AFRL-IF-RS-TR-2007-198
Final Technical Report
September 2007

DECISION SUPPORT UNDER UNCERTAINTY USING EXPLORATORY MULTISIMULATION WITH MULTIRESOLUTION MULTISTAGE MODELS

Auburn University

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

NOTICE AND SIGNATURE PAGE

AFRL-IF-RS-TR-2007-198 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/	/s/
DAWN TREVISANI	JAMES W. CUSACK
Work Unit Manager	Chief, Information Systems Division
	Information Directorate

REPORT DOCUMENTATION PAGE	*Form Approved* **OMB No. 0704-0188**

1. REPORT DATE *(DD-MM-YYYY)* SEP 2007	**2. REPORT TYPE** Final	**3. DATES COVERED** *(From - To)* Jun 06 – Jun 07

4. TITLE AND SUBTITLE DECISION SUPPORT UNDER UNCERTAINTY USING EXPLORATORY MULTISIMULATION WITH MULTIRESOLUTION MULTISTAGE MODELS	**5a. CONTRACT NUMBER** FA8750-06-1-0200
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER 62702F
6. AUTHOR(S) Alvin Lim	**5d. PROJECT NUMBER** 459S
	5e. TASK NUMBER N6
	5f. WORK UNIT NUMBER 03
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Auburn University 107 Samford Hall Auburn AL 36849-0001	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/IFSB 525 Brooks Rd Rome NY 13441-4505	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSORING/MONITORING AGENCY REPORT NUMBER AFRL-IF-RS-TR-2007-198

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA# AFRL-07-0087

13. SUPPLEMENTARY NOTES

14. ABSTRACT
In most realistic simulation-supported decision making situations, the nature of the problem changes as the simulation unfolds. Initial parameters, as well as models can be irrelevant under emergent conditions. Furthermore, our knowledge about the problem being study may not be captured by any single model or experiment. Therefore, adaptivity in simulations and scenarios is necessary to deal with emergent conditions and for evolving systems in a flexible manner. Dealing with uncertainty is paramount to analyzing complex evolving phenomena. This effort involved (1) a new advanced simulation methodology, called exploratory multisimulation to promote problem space exploration, as opposed to traditional solution space exploration concept to deal with challenges pertaining to irregular and asymmetric warfare and (2) a symbiotic multisimulation-based decision support system based on the Naturalistic Decision Making paradigm. The decision support system illustrates how situational awareness can be enhanced by intelligent agents with perception, understanding, and anticipation capabilities to support multisimulation-based exploration of this problem area.

15. SUBJECT TERMS
Exploratory Multisimulation, symbiotic multisimulation, Naturalistic Decision Making, decision support system, multiresolution modeling

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Dawn Trevisani
a. REPORT U	**b. ABSTRACT** U	**c. THIS PAGE** U	UL	58	**19b. TELEPHONE NUMBER** (*Include area code)*

Table of Contents

List of Figures

1. Introduction

Strategy problems are typically characterized by significant uncertainty. Proper simulation-based decision support methodologies that are consistent with the way experts use their experience to make decisions in field settings could improve modeling Course of Actions (COAs), simulating them faster than real time, and then performing COA analysis. Exploring the effectiveness of alternative COAs at the tactical and operational levels requires dynamic updating, branching, and simultaneous execution of simulations, potentially at different levels of resolution. Consider, for instance, the following scenario.

Motivating Scenario – Adaptive experience management in strategic leadership training: *An inspection team under the command of the team of participants is at a weapons storage site in a fictional city. The inspection team discovers that weapons from the site are missing and that a hostile crowd is forming around them. As the inspection team radios for help, the members of the command staff must prepare and launch a rescue operation. Evidence begins to mount that the weapons were stolen by paramilitary troops who are motivating the hostile crowd. As additional paramilitary troops stream into the town, the command staff must overcome a series of obstacles in order to rescue the inspection team without incident or injury*.

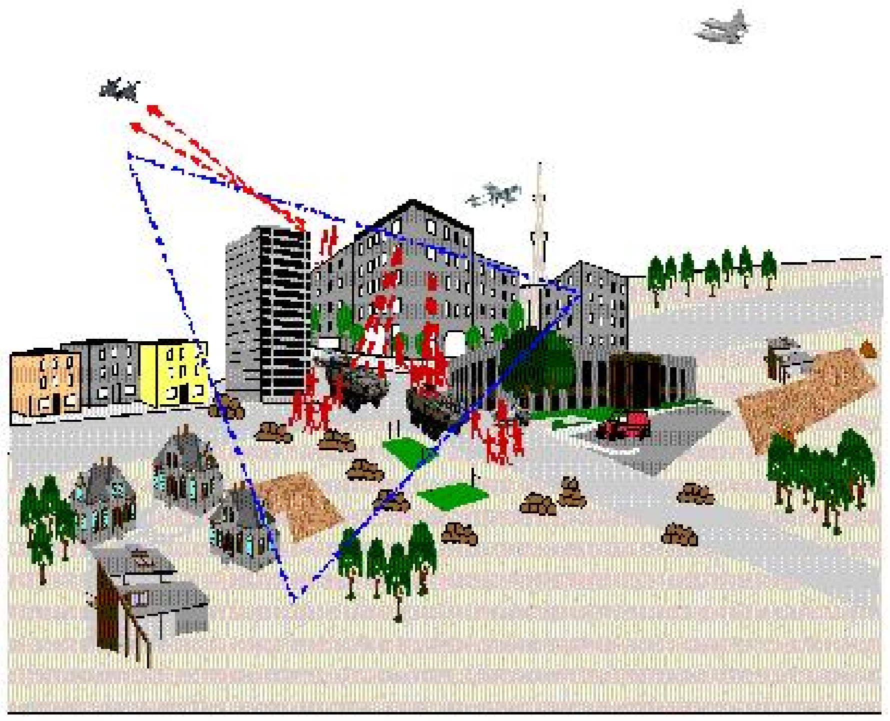


Figure 1 - Asymmetric and Irregular Conflicts

Given the impossibility of foreseeing all moves in a conflict, decision makers need to consider contingency models on demand to explore and determine plausible consequences of their COAs if the assumptions underlying these assumptions turn out to be true. The evolution of the environment in which the decision makers operate could require identifying and bringing one or more new family of models that are consistent with the observed state of the context. For instance, the decision maker may foresee that the original plan may result in civilian casualty within the new observed context. Hence, experimenting with contingency models in real-time on demand would be critical, as decision support is considered for unstructured problems with the characteristics of (1) deep uncertainty, (2) dynamic environments, (3) shifting, ill-defined, and competing goals, (4) action/feedback loops, and (5) time stress.

The major challenges pertaining to decision making for such asymmetric and irregular environments as depicted in the scenario are as follows:

1. For most realistic problems, the nature of the problem changes as the simulation unfolds. Initial parameters, as well as models can be irrelevant under emergent conditions. Relevant models need to be identified and instantiated to continue exploration. Manual exploration is not cost effective and realistic within a large problem state space.
2. Our knowledge about the problem being studied may not be captured by any single model or experiment. Instead, the available knowledge is viewed as being contained in the collection of all possible modeling experiments that are plausible given what is known and what is learned.
3. Dealing with uncertainty is paramount to analyzing complex evolving phenomena. Adaptivity in simulations and scenarios is necessary to deal with emergent conditions for evolving systems in a flexible manner.

This study presents a strategy for developing decision support systems. The two major contributions of the framework and the environment developed are (1) the introduction of a new simulation strategy, called exploratory multisimulation, for the simulation of several aspects of a phenomena and (2) the framing of the decision making process and associated cognitive mechanisms around the *Naturalistic Decision Making* (NDM) (Klein 2007) paradigm. Exploratory multisimulation could be a very useful framework for optimization of the solution across multiple problem spaces, particularly for asymmetric warfare. As several problem spaces are explored and simulated simultaneously, the optimal solution for each problem space can be compared and the overall optimal solution for multiple possible problem spaces can be derived through techniques yet to be determined.

The rest of the report is organized as follows. In section 2, an overview of human decision making is presented to elaborate on the rationale for choosing NDM as the decision making process of choice. Section 3 discusses the themes, strategy, and the supporting techniques for addressing the challenges above. Section 4 introduces our approach in terms of the strategy and the supporting technology outlined in section 3. To demonstrate the utility of the proposed framework, a case study is performed and presented in section 5. The lessons learned and experienced gained during the design and implementation of the case study resulted in requirements and design principles for next generation simulation-based decision support system. These requirements are delineated in section 6. Finally, in section 6, we summarize our findings and conclude by discussing potential avenues of further research.

2. Background in Human Decision Making

Decision science contributes both to the understanding of decision making and developing tools, environments, and supporting methods to assist that decision making (Davis et al. 2005). Decision-making is viewed as a process that entails two distinct activities. The first one is to decide what state of affairs is desired and second how this state will be achieved. In modern decision science, there are mainly three decision-making styles.

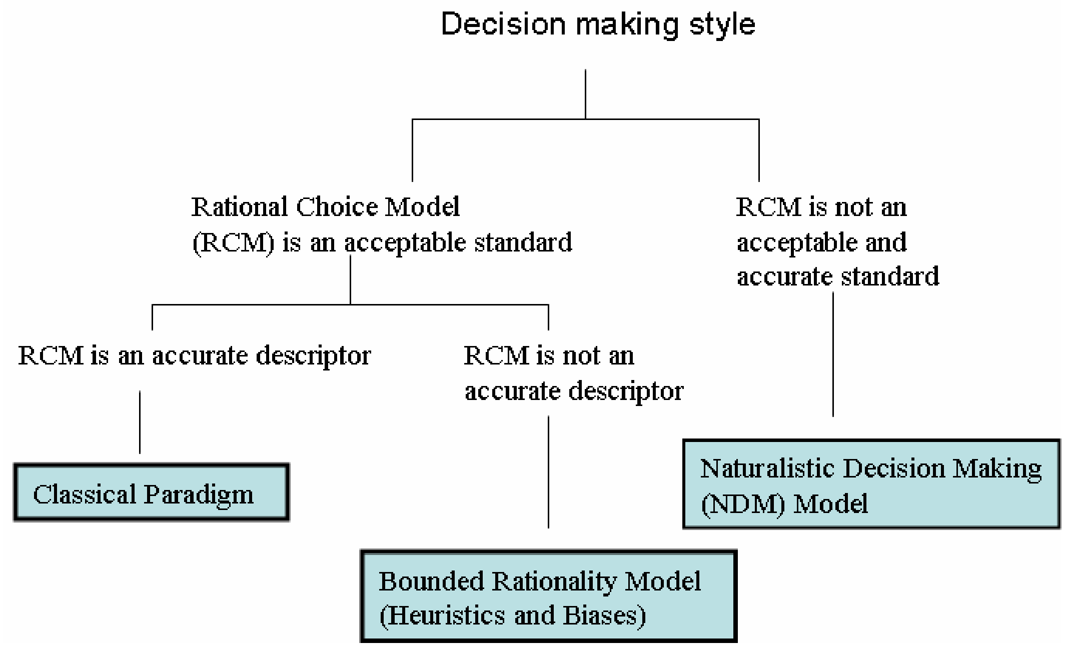

Decision making style
Rational Choice Model (RCM) is an acceptable standard
RCM is not an acceptable and accurate standard
RCM is an accurate descriptor
RCM is not an accurate descriptor
Classical Paradigm
Naturalistic Decision Making (NDM) Model
Bounded Rationality Model (Heuristics and Biases)

Figure 2 - Decision Making Styles

- *Rational Choice Model (RCM):* This model of decision-making emerged in such diverse fields as economics, political science, management science, and operation research. Von Neumeann and Morgenstern (1953) introduced the idea that rational choice should maximize *expected subjective utility*. From the perspective of game theory, this classic approach to decision analysis can be viewed as an analytical approach that optimizes the outcome of a decision. Building on the rationality principle, game theory has been applied to various problems (Geyer and Zouven 1998, Shubik 1964). However, evidence exists that classical game theory fails in cases where opponents have different value systems (Knight et al. 1991). Different types of game theories (e.g., sequential games, repeated games (Banks and Sundaram 1990, Leimar 1997) differential games, evolutionary games, and hypergames (Fraser and Hipel 1984), have been applied in the context of RCM.

- *Bounded Rationality (BR)*: In making decisions, humans operate within a complex and often changing environment with limited cognitive capabilities, time, and other resources. Hence, decision-making is only rational within the bounds imposed on decision makers (Simon 1982). Kahneman and Tversky (1974) identified a number of heuristics and biases that humans use to make decisions. These studies aim to bring classical and analytic decision theorists into conformity with findings in cognitive psychology. The premise of bounded rationally is based on the observation that heuristics (Davis et al. 2005) often yield cost-effective compared to classical methods in terms of time and mental effort. Furthermore, changes in the environment will cause the judgment to be obsolete.
- *Naturalistic Decision-making (NDM)*: The empirical work of Gary Klein (1997) on expert behavior in high-pressure environments resulted in a new school of thought in decision-making. The NDM paradigm argues that people assess situations by using prior experience and knowledge. Furthermore, unlike RCM and BR decision-making styles, in NDM situation assessment is considered to be more important compared to option generation. Hence, the approach is to perform pattern matching to match observed problem facets to the mental model of the problem formed by the decision maker. Sokolowski (2003) discusses the application of NDM for agent supported decision-making.

Decision-making involves making tradeoffs among competing attributes or goals, analyzing complex situations within constraints of time and resources, projecting into future state of the environment despite uncertainty, and making judgments, even if they are heuristic (Zachary 1998). The evolution of decision-making theory can be viewed as a steady withdrawal from the rational choice model to bounded rationality, and most recently to naturalistic decision-making (NDM) theory. While rational choice model (Parsons and Wooldridge 2002) involves the maximization or optimization of the expected utilities, bounded rationality emphasizes the constraints of time, resources, and cognitive capacities. Bounded rationality worldview involves the use of heuristics and biases (Tversky and Kahneman 1974) to capture cognitive shortcuts used in decision-making. Naturalistic decision-making, on the other hand, is based on the premise that humans assess situations by using prior experience. Zsambok (1997) argues that situation assessment and experience-based decision-making is more appropriate than option generation under conditions that involve uncertain and dynamic environments, shifting or competing goals, time stress, and ill-structured problems. Note that decision-making styles can shift between analytic, heuristic, and experience-based several times within a single problem (Hamm 1988). Furthermore, (Hammond 1986) demonstrates that task features, such as complexity of the task structure, ambiguity, and form of representation, determine the decision-making style.

The nature of the decision style further imposes constraints on the decision-making models within a multi-model. Table 1 depicts the three main decision styles along with the problem domain characteristics they target. For instance, the RCM style provides an acceptable and accurate framework for problems in which actors, their preferences, utilities for actions, and the outcomes are well-defined. The problem is expected to be stable, and the number of options and players are small. Furthermore, the cognitive limitations of the decision maker and the lack of resources are not considered to inhibiting factors in decision-making. Having decision-making tools that enable formal specification of the structure of decision-making problem is feasible under these conditions. Therefore, interactive tools that provide graphical facilities to capture options, preferences, utilities etc. can be useful. On the other hand, NDM decision-making style is introduced for problem domains that are ill-defined. The level of uncertainty in the environment leads to shifting and possibly competing goals.

Decision-making style	**Problem Domain Characteristics**	**Tool Design Features**
Rational Choice Model	1- Well-defined problems 2- Low uncertainty 3- Stable environment 4- Small number of players and options 5- Time is not a parameter/factor	a- High-level design templates for various recurring problems b- Graphical interfaces for specifying utilities, actors, preferences, and outcomes
Bounded Rationality	1- Resource limitations (cognitive, computational etc.) 2- Time stress is a factor 3- Medium level certainty 4- Incomplete information about the environment	a- Models that encode heuristics and biases such as availability, representativeness, and anchoring and adjustment heuristics (Davis et al. 2005)
Naturalistic Decision Making	1- Ill-structured problems 2-Uncertain, dynamic environments 3- Shifting, ill-defined, competing goals 4- Action/feedback loops 5- Time stress and high stakes 6- Multiple players 7- Organizational goals and norms are factors. (Szambok 1997)	a- Perceiving situations in an environment b- Matching perceptions against learned experiences c- Understanding the overall situation via comprehension mechanisms d- Exploring possible outcomes by emulating mental simulation d- Anticipating future state(s) of the environment before making a decision

Table 1 - Features of Decision-making Styles

The characteristics of the domain are common in decision-making environments where there is a time stress, high stakes, and continuous action/feedback loops. To support experts in making decisions in such environments, a decision-support system needs to provide facilities to augment pattern matching for situation recognition, understanding of the overall situation from the perceived disconnected elements, and make projection to potential future states. The projection phase simply involves tool support for mental simulation of the plausible actions.

3. A Framework for Advanced Decision Science

Developing a framework for next generation decision support systems (DSS) requires understanding the themes, strategies, and available supporting technology. The themes refer to worldview or perspective that underlies the principles, which govern the operation of DSS in uncertain and evolving environments. Strategies are the available methodologies that are capable of addressing the challenges pertaining to such systems. The supporting technology refers to technical methods and infrastructures that facilitate the realization of selected strategies.

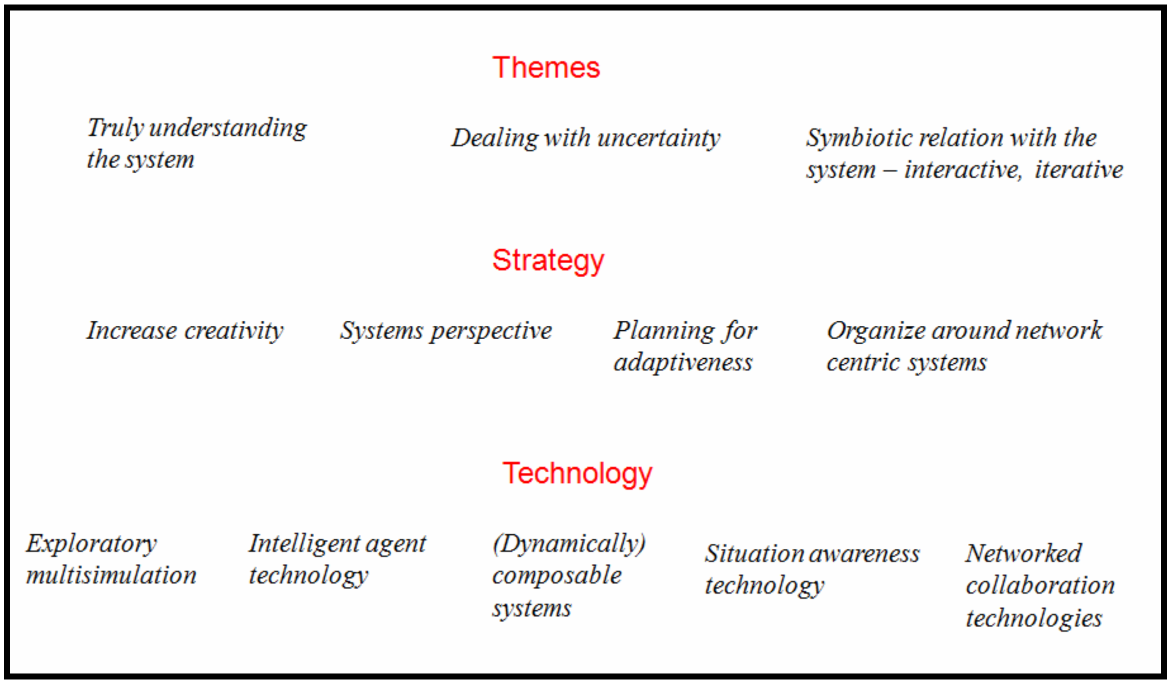

Themes
Truly understanding the system
Dealing with uncertainty
Symbiotic relation with the system – interactive, iterative
Strategy
Increase creativity
Systems perspective
Planning for adaptiveness
Organize around network centric systems
Technology
Exploratory multisimulation
Intelligent agent technology
(Dynamically) composable systems
Situation awareness technology
Networked collaboration technologies

Figure 3 - A Framework for Advanced Decision Science

Figure 3 lays out the relationship between the technology, strategy, and themes underlying the proposed approach. Next, we briefly discuss the major elements of the above framework as they pertain to the DSS approach advocated and developed in this study.

3.1 Dealing with Uncertainty

Uncertainty in unstructured and dynamically evolving systems or environments along with time stress in making decisions precludes optimizing the outcome of decisions in most realistic scenarios. Instead of seeking to predict the effects on a system of various alternatives and then “optimizing” choice, it may be far better to recognize that meaningful prediction is not plausible and that we should instead be seeking strategies that are flexible, adaptive, and robust. Dealing with uncertainty puts emphasis on modular capabilities that are usable in various alternative ways with associated assembly capabilities. In military terms, this is in contrast with developing units, equipment, plans, doctrine designed to do extremely well in a specific scenario.

3.2 Near Real-time Decision Making with Symbiotic DSS

Real-time decision making requires models and tools that allow interaction with the system of interest. A symbiotic DSS is defined as one that interacts with the physical system in a mutually beneficial way. It is highly adaptive – not only performs “what if” experiments to control the physical system, but also accepts and responds to data from the physical system.

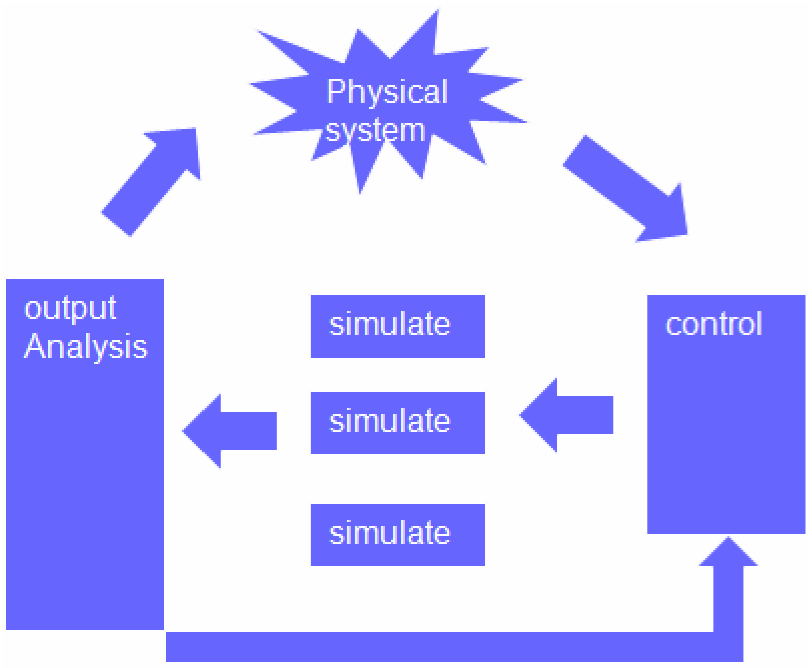


Figure 4 - Symbiotic Decision Support

Figure 4 presents the major components of a Symbiotic DSS, which is based on simulation-based analysis of alternative COAs. Note, however, the approach applies alternative decision making strategies that include methods other than simulation. The control component is responsible for assessing the state of the environment and configures the decision making component (e.g., simulation) on demand based on its perception of the environment as well as the requests initiated by the output analysis component.

3.3 Intelligent Agent Technology

Software agents are entities that (1) are capable of acting in purely software and/or mixed hardware/software environments (2) can communicate directly with other agents, (3) are driven by a set of goals, objectives and tendencies, (4) possess skills to offer services, (5) perceive its environment, and (6) can generate autonomous behavior that tends toward satisfying its objectives. An overview of additional views is documented in Murch and Johnson (1998). Furthermore, we assume that the environment will be not-accessible (versus accessible), stochastic (versus deterministic), dynamic (versus static), sequential (versus episodic), and continuous (versus discrete) to represent the environments specified in the last section for realistic decision-making problems. In this context, we understand agents as autonomous software modules with perception and social ability to perform goal-directed knowledge processing over time, on behalf of humans or other agents in software and physical environments. When agents operate in physical environments, they can be used in the implementation of intelligent machines and intelligent systems and can interact with their environment by sensors and effectors. The core knowledge processing abilities of agents include: reasoning, motivation, planning, and decision-making. The factors that may affect decision-making of agents, such as personality, emotions, and cultural backgrounds can also be embedded within agents. Additional abilities of agents are needed to increase their intelligence and trustworthiness. Abilities to make agents *intelligent* include anticipation (pro-activeness), understanding, learning, and communication in natural and body language. In this chapter, we advocate the use of (1) practical situation-aware agents that diagnose the situation via perception, understanding, and anticipation capabilities and (2) agents that facilitate simulation-based analysis of alternative COAs.

3.4 Exploratory Multisimulation

Common efforts to deal with uncertainty in complex adaptive social systems using simulation include Exploratory Modeling, Exploratory Analysis, and Multisimulation. The methods employed by these fields are different from traditional simulation techniques in engineering which depend on accurate and valid models of a physical system to make predictions about its behavior. Central to the practice of each approach is the notion of a plausible model. In (Bankes, 1998) plausible models are described as predicting “how the system would behave if the assumptions the model is based on were true”. The behavior of such models can provide insights that lead to improved avenues of inquiry. This is different from an authoritative model that accurately represents the system in question. Because of the presence of uncertainty, there may be many plausible models that could represent a system (Bankes, 1993). Similarly, knowledge of the system constrains the set of plausible models. Therefore, all three of the previously mentioned fields of simulation experiment with ensembles of models as experimentation with a single plausible model would be just as likely deceptive as informative.

Among a set of plausible models, variation occurs according to input uncertainty and structural uncertainty (Davis, 2000). Input uncertainty which deals with the possible values that model input factors may assume can be further subdivided into model uncertainty and parameter uncertainty (Henderson, 2003). Model uncertainty is dependent on the choice of distribution function for a particular input whereas parameter uncertainty relates to the choice of parameters that govern the shape of those distributions. Classical experimental design techniques can be useful in dealing with input uncertainty (Barton, 2004) provided the factors to be examined are relatively few in number and their interactions are linear. Structural uncertainty can be much more challenging to deal with. This includes such things as the actual choice of variables used to represent a system. It may arise from difficulty in analyzing a system or even from differences of expert opinion.

Because of the variation that can occur across a set of plausible models, methods for consistently representing them have been of interest. In the fields Exploratory Modeling and Exploratory Analysis, there has generally been a focus toward parameterizing both kinds of uncertainties. In the field of Multisimulation, there has been an emphasis toward the use of the multimodel formalism. An examination of the kinds of models described by this formalism can be found in (Yilmaz et al., 2006). In general, the multimodel formalism provides a specification of how models may vary in relation to each other. This specification is necessary to permit dynamic replacement and updating of models which is an area of specific interest in Multisimulation.

Multiresolution Modeling (MRM), for instance, is a type of multimodel that is of particular importance in decision support. MRM has been studied and used extensively in all three fields. MRM is described in (Davis & Bigelow, 1999) as "constructing a model or family of models that describe consistently the same system or process at different levels of resolution". The authors outline several motivations for MRM. The two most important of these are economy and dealing with chaos. In the first case, exploring a set of models in a high level way before examining certain phenomena in more detail can make efficient use of computational resources. Chaos on the other hand means that minute details may result in significantly different end states of a system, even when the initial states of two models appear reasonably similar. Davis and Bigelow point out that chaos is common in systems with non-linear dynamics. MRM goes against the usual technique of modeling a system from the bottom up. It suggests that models with varying amounts of detail can be informative in dealing with chaos and thus mutually calibrating. This notion is also heavily discussed in (Bigelow & Davis, 2002). One issue raised there is that of aggregation and disaggregation of input factors. In order to maintain consistency between levels of resolution, there needs to be an appropriate mapping between factors in low resolution models to their constituents in high resolution models.

While multimodel techniques like MRM are helpful in dealing with the variation among a set of plausible models, the sheer size of a typical ensemble can pose equally difficult challenges in terms of computational resources. Knowledge of the system under inquiry can reduce the size of the model ensemble, but even with significant knowledge, the number of plausible models could still be infinite (Bankes, 93). In light of this problem, effective techniques for sampling a model ensemble are of interest. Bankes asserts that the choice of technique is largely dependent on the purpose of the study and for models that maximize certain outputs of interest, a search strategy should be used.

Additional insight into the problem of sampling model ensembles can be found in (Davis, 2000). The author makes a distinction between controllable and uncontrollable input factors of a model. In general, controllable inputs are considered to be the factors that a decision maker could actually change within the physical system in response to observations made within a simulation study. Uncontrollable factors represent environmental conditions within which a proposed system operates. This distinction is also mentioned on page 620 of (Law, 2007). For each type of factor, a different sampling technique is appropriate. Davis asserts that for uncontrollable inputs, Probabilistic Exploration should be used. With Probabilistic Exploration an attempt is made to identify a suitable distribution function for each uncontrollable input. The choice of function is often based on the subjective knowledge of the analyst. For the controllable input factors of a model, Davis suggests a Parametric Exploration strategy. In Exploratory Analysis, Parametric Exploration involves a combination of classical experimental design with software visualization tools. Using such tools, an analyst is capable of identifying significant interactions between factors in a matter of minutes. Davis refers to this strategy of using different techniques to explore controllable and uncontrollable inputs as Hybrid Exploration. Exploratory Modeling also involves the use of Hybrid Exploration but differs in terms of how controllable inputs are handled. This difference is motivated by the goals of Exploratory Modeling which are often concerned with the evaluation of several potential system designs (Davis et al., 2007).

4. A Strategy for Augmenting NDM

As indicated in Figure 5, development of a DSS requires understanding the process involved in making a decision along with the cognition and reasoning mechanisms taking place during decision making. Augmenting the decision making process of experts using proper computational methods needs to be predicated on the decision making style that is consistent with the problem domain characteristics. This study examines the NDM lifecycle and presents methods to support the phases and activities that are specific to NDM.

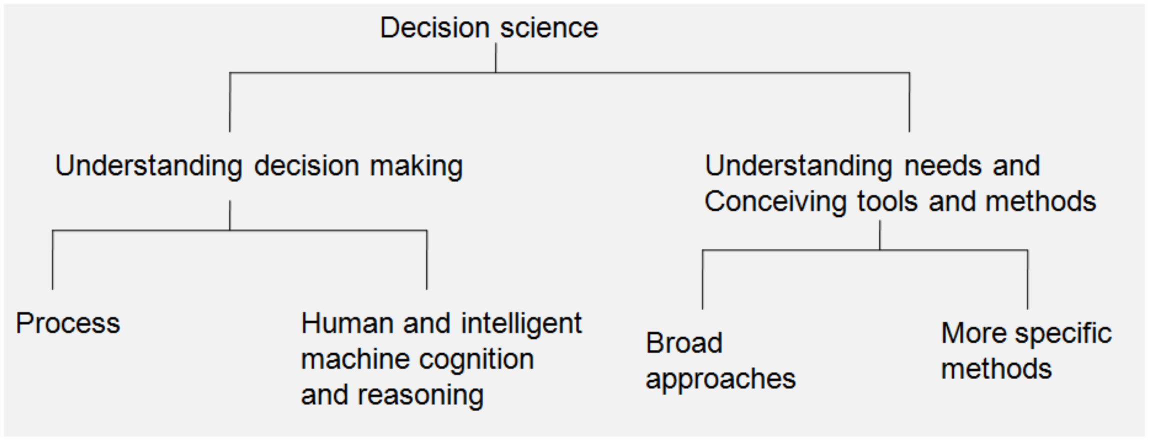

Decision science
Understanding decision making
Understanding needs and Conceiving tools and methods
Process
Human and intelligent machine cognition and reasoning
Broad approaches
More specific methods

Figure 5 - Elements of Decision Science (Davis et al. 2005)

4.1 The Process

The proposed model, which is based on Recognition Primed Decision Model, is an example of Naturalistic Decision Model (NDM), and it attempts to emulate what people actually do under conditions of time pressure, ambiguous information, and changing conditions. According to the architecture, the sensory input is processed by the *experience the situation* component to perceive the elements of the situation. If the situation is prototypical, the NDM submodel instantiates a skeleton mental model, from which expectancies and goals can be derived. Simple if-then rules can be used to derive plausible actions based on goal-action pairs. These goal-action pairs are based on prior experience, and they are encoded within the mental model. If the observed situation and perceived inputs are not categorized to be prototypical, then a diagnosis (i.e., pattern matching) procedure that synthesizes the features of the percepts to causal factors is enacted to facilitate comprehending the situation until a prototypical or analog case is identified. The exploration phase of the life cycle requires evaluating the selected action. Humans often perform mental simulation of the possible outcomes if and when the decision is implemented. In our system, the evaluation is performed via multisimulation. If the action is found to be irrelevant to the goal as a result of the projection or mental simulation, the mental model is further revised to either update the goal or identify a different action.

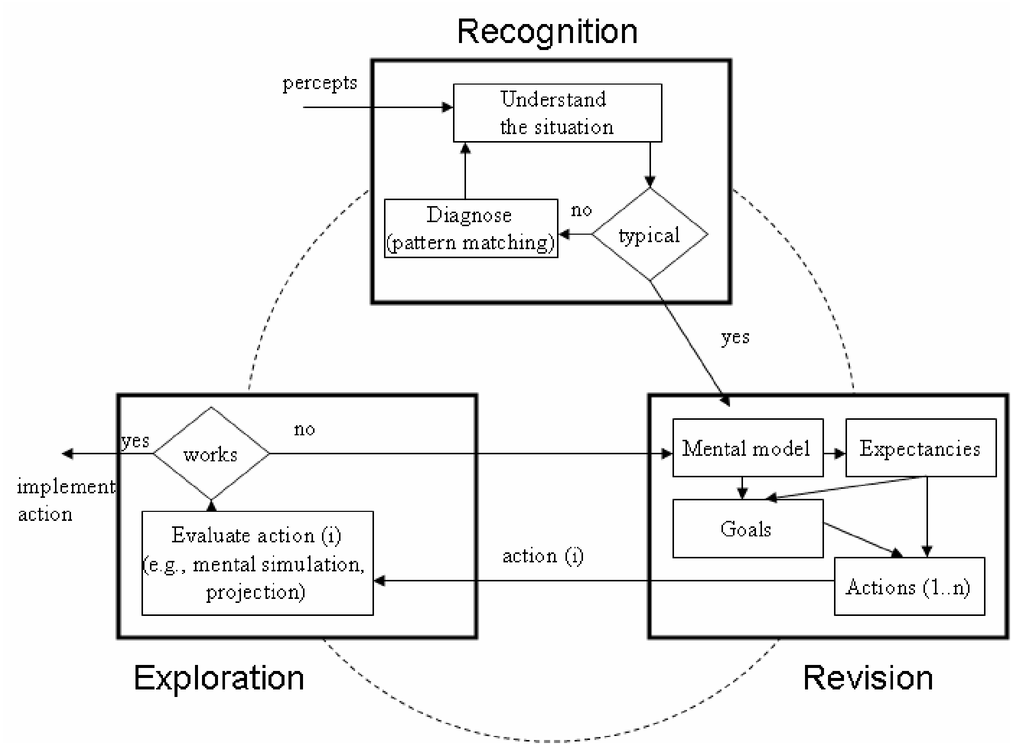


Figure 6 - Decision Making Lifecycle

4.2 The Role of Situation Awareness in Decision Support

Endsley (1995) defines situation awareness as the perception of elements in a particular environment within time and space, the comprehension of their meaning and the projection of their status in the near future. Situation awareness, as depicted here, provides a set of mechanisms that enable attention to cues in the environment, and expectancies regarding future states. In realistic settings, establishing an ongoing awareness and understanding of important situation components pose the major task of the decision maker. Therefore, situation awareness is the primary basis of the decision making process in experience-based decision making processes (i.e., Naturalistic Decision Making).

The recognition, revision, and exploration phases of the situation awareness, shown in Figure 7, suggest three main functional areas that revolve around a mental model of the problem domain. More specifically, a well-defined mental model provides:

1. knowledge about the concepts, attributes, associations, and constraints that pertain to the application domain,
2. a mechanism that facilitates integration of domain elements to form an understanding of the situation, and
3. a mechanism to project to a future state of the environment given the current state, selected action, and the knowledge about the dynamics of the environment.

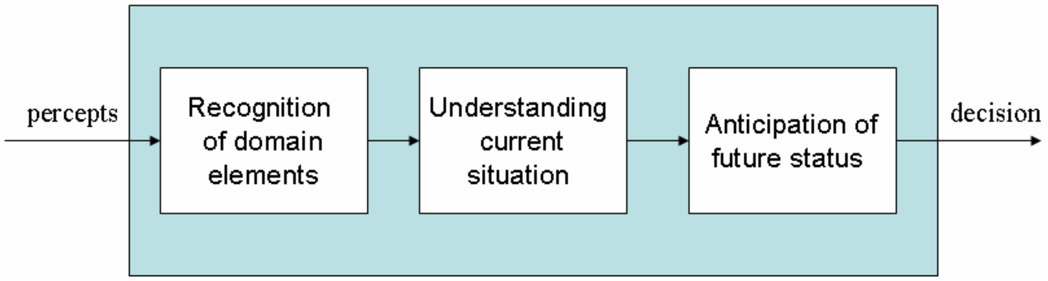

percepts
Recognition of domain elements
Understanding current situation
Anticipation of future status
decision

Figure 7 - Elements of Situation Awareness

Situation awareness is an important cognitive skill that is essential for expert performance in any field involving complexity, dynamism, uncertainty, and risk. The failure to perceive a situation correctly may lead to faulty understanding. Ultimately, this misunderstanding may degrade an individual's ability to predict future states and engage in effective decision making (Gaba and Howard 1995). It is therefore an essential part of the NDM.

Perception

The way we perceive reality affects our feelings, decisions, and actions. Since Plato's allegory of the cave that he explained in Book 7 of the Republic, it is well known that perception is very important (Bloom 1968). Wikipedia encyclopedia explains philosophy of perception as follows:

"The ***philosophy of perception*** *concerns how mental processes and symbols depend on the world internal and external to the perceiver. Our perception of the external world begins with the senses, which lead us to generate empirical concepts representing the world around us, within a mental framework relating new concepts to preexisting ones. Because perception leads to an individual's impression of the world, its study may be important for those interested in better understanding communication, self, id, ego –even reality" (Wikipedia-Phi-per 2004).*

There are two types of perception, i.e., external and internal perceptions. Philosophy of perception is concerned with external or sensory perception.

- "*External* or sensory *perception*, tells us about the world outside our bodies. Using our senses of sight, hearing, touch, smell, and taste, we discover colors, sounds, textures, etc. of the world at large.
- *Internal perception* tells us what's going on in our bodies. We can sense where our limbs are, whether we're sitting or standing; we can also sense whether we are hungry, or tired, and so forth." (Wikipedia-Phi-per 2004).

Both types of perceptions can involve thought processes. *Introspection* is the detailed mental self-examination of feelings, thoughts, and motives.

“In psychology and the cognitive sciences, ***perception*** *is the process of acquiring, interpreting, selecting, and organizing sensory information. Methods of studying perception range from essentially biological or physiological approaches, through psychological approaches to the often abstract ‘thought-experiments’ of mental philosophy” (Wikipedia-Per 2004).*

A categorization of perception is given in Table 2. Perception of an entity at a time *t* gives an image of it at that time. At time *t*, we can refer to the perception as the current perception (or current image), if there is only one perception.

	Current images of	
	Past or current state	**Future state**
Others (people and/or events)	Perceived image of others and events	Behavioral anticipation of others and events
Self (decision maker(s), supporters, followers, and/or events related with one’s own side)	Perceived image of self and/or events related with one’s own side	Behavioral anticipation of self and/or events related with one’s own side

Table 2 - Categories of Perception

However, at a time *t*, based on the perspective, there may be different interpretations of an entity, hence several perceptions. From now on, for the sake of simplicity, unless it is specified otherwise, current perception (or current image) is considered to be unique. Current image can refer to external perceptions; hence it can be about others (people, groups, nations, events, facts, etc.). When current image refers to internal perceptions, then it is about self (or own group of decision makers, supporters, followers; and/or events related with one’s own side.) Current image may refer to past, current, or future states. There can be several current images, at different times t_i, i=1, 2, 3, …, n; until *future* becomes current. This is similar to for example, seven day meteorological forecasts. At each day, there can be a forecast of a certain day until that day. And due to the variability of meteorological conditions, the forecasts may be different. When that specific day occurs, what we experience is the current image of the current state. If we are interested to interpret past events, current image(s) of a certain *past* may be defined. However, there can be several images of a certain past based on the points of views of the people involved. Current images of (past, current, or future states) can reflect possibly different interpretations of the current perceptions. Hence, especially in a conflict situation, the opponents may even have antagonistic interpretations of the same situation. Furthermore, emotions such as anger affect the disposition of the decision makers.

Understanding

Understanding or comprehension of the situation is based on synthesizing the perceived disjoint elements to form a coherent representation of the entity, the elements of which are observed. For instance, the tactical commander of a military unit needs to comprehend that the appearance of enemy aligned in a specific pattern and in a particular location depicts certain specific objectives.

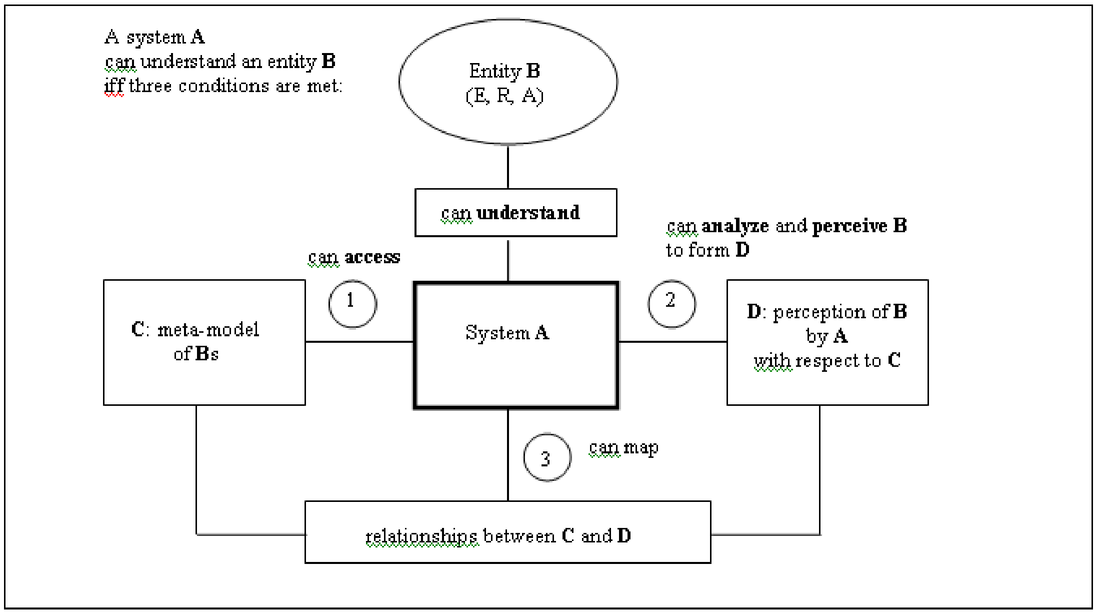


Figure 8 - Model of Understanding

A system **A** can understand an entity **B** (**E**ntity, **R**elation, **A**ttribute) iff three conditions are satisfied:

- 1. **A** can access **C**, a meta-model of Bs.
- 2. **A** can analyze and perceive **B** to generate **D**. (D is a perception of B by A with respect to C.)
- **A** can map relationships between **C** and **D** for existing and non-existing features in **C** and/or **D** to generate result (or product) of understanding process.

A Model of Anticipation in Decision Making

Anticipation is an important characteristic of intelligence. Pro-active behavior requires anticipatory abilities. Without anticipation a system can only be reactive; a dead frog can also be reactive. A seminal work on anticipatory systems is the one written by Rosen (1985). A brief introduction to and serious concerns about anticipation follows:

"Strictly speaking, an anticipatory system is one in which present change of state depends upon future circumstances, rather than merely on the present or past. As such, anticipation

has routinely been excluded from any kind of systematic study, on the grounds that it violates the causal foundation on which all of theoretical science must rest, and on the grounds that it introduces a telic element which is scientifically unacceptable. Nevertheless, biology is replete with situations in which organisms can generate and maintain internal predictive models of themselves and their environments, and utilize the predictions of these models about the future for purpose of control in the present. Many of the unique properties of organisms can really be understood only if these internal models are taken into account. Thus, the concept of a system with an internal predictive model seemed to offer a way to study anticipatory systems in a scientifically rigorous way" (Rosen 1985, from forward).

Dubois started a series of conferences on anticipatory systems (Dubois 1998, 2000). A systematic review of 12 definitions of anticipation is available from BISC-SIG with the following warning:

"The following 12 definitions, or descriptions, of anticipation should be understood as working hypotheses. It is hoped and expected that the knowledge community of those interested in anticipation will eventually refine these definitions and suggest new ones in order to facilitate a better understanding of what anticipation is and its importance for the survival of living systems" (BISC-SIG 2004).

An important aspect from the point of view of BISC-SIG is the emphasis on soft computing requirements in anticipation. Perception ability is a required characteristic of agents. Hence, they can be designed to perceive current state of self and others. They can also be designed to create current image(s) of future state(s).
An anticipatory system is a system whose next state depends on its current state as well as the current image(s) of its future state(s). This definition is a radical departure from the original definition given by Rosen (1985):

"An anticipatory system is a system determined by a future state. The cause lies in the future."

Nonetheless, our definition is in line with the following definition also given by Rosen:

"An anticipatory system is a system containing a predictive model of itself and/or of its environment that allows it to change state at an instant in accord with the model's predictions pertaining to a later instant" (Rosen 1985).

However, we would like to stress the distinction on dependency of next states on current image(s) of future state(s) rather than the future value of the states. Perception requires mechanisms that enable interpretive capabilities. Perception invariably involves sensory qualities, and introspection entails accessing sensations and perceptions the agent would introspect. Perceptions are derived as a result of interpretation of sensory inputs within the context of the current world and agent's self model. The prototype inference, orientation accounting, and situational classification mechanisms (Sallach 2003) could be used to realize the interpretation capabilities of an agent. The interpretation process results in perceptions. An anticipatory agent needs to deliberate upon perceptions through introspection and reflection to anticipate.

Introspection is deliberate and attentive because higher-order intentional states are themselves attentive and deliberate. An introspective agent should have access mechanisms to its internal representation, operations, behavioral potentials, and beliefs about its context. Reflection uses the introspective mechanisms to deliberate its situation in relation to the embedding environmental context. These features collectively result in anticipation capabilities that orient and situate an agent for accurate future projections. Figure 6 presents interpretation and introspection as critical components within the micro-architecture of an anticipatory agent.

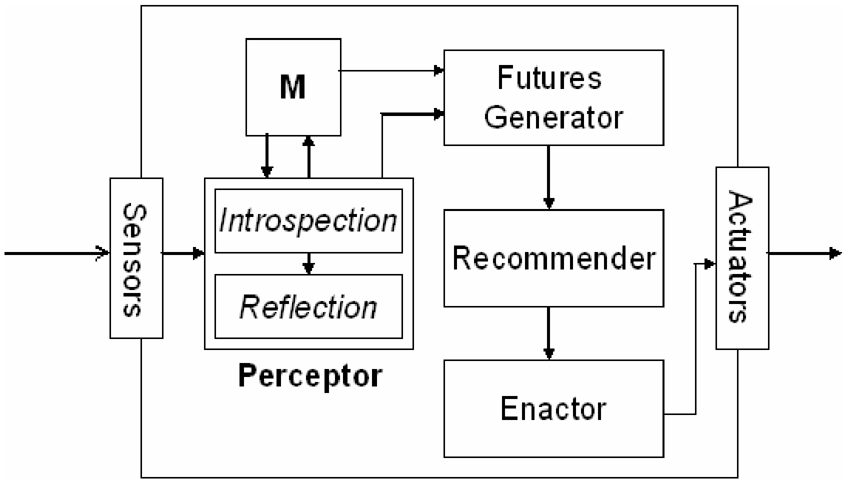


Figure 9 - Basic Components for Anticipatory Agents

A computationally anticipatory agent needs to incorporate interpretation facilities as a precursor to (1) comprehend and draw accurate inferences about the world, (2) have social pragmatism by considering the likely responses of others in its context in response to a communication or act, and (3) have situational definition (Sallach 2003) as a direct input to action recommendation. An anticipatory agent uses a domain model **M**, as the internal representation of the environment and agent’s self in order to project to the future. The model and the anticipation that results from the introspection and reflection processes are used to derive a number of realities by the futures generator. The generator is a function that maps environmental parameters and past vector of states onto a set of future states of the environment. Naturally, an inductive process would be used to realize the function, as the generation of future plausible realities (environmental contexts) results in a set of new models that vary from each other based on assumptions on different plausible events or possible interactions between the environment and the agent itself. This perspective is consistent with the definition of anticipation process that is given in (BISC-SIG 2004). According to the definition, anticipation (1) is a realization within the domain of possibilities and/or (2) involves the generation of a multitude of dynamic models and the resolution of their conflict. As such, the recommender subsystem is responsible for evaluating alternative anticipated models and to decide on choosing a specific strategy based on the goals and motivations of the agent. Next, a recommender system should select a desirable future state upon which the agent would make decisions and react using its enactor component.

Developing anticipatory agents with runtime recommenders is difficult, because interpretation of emergent conditions requires mining the state of the simulation to recognize situations within the domain theory (schema) of an application. That is plausible and desirable future states need to be qualified based on the motives and goals of the agents. Learning takes place as recommendations are made. Adaptive models that assume certain discernible patterns in the recommendations may be used to discover situations and associated relevant models so as to reinforce qualification of specific future states based on previous experience. Various domain specific representational issues and inadequacies make this very difficult for particular applications. One form of representational inadequacy pertains to intrinsic difficulty of determining (and utilizing) the features that are potentially relevant for model selection. Another form of representational inadequacy involves on deciding the right level of detail. A major difference between traditional deliberative agents and an anticipatory agent is that an anticipatory agent makes guesses about the future state of the environment to guide its behavior, whereas conventional deliberative agents make their decisions based on the observed conditions within the current context.

4.3 The Role of Exploratory Multisimulation in Decision Support

Multisimulation (or multisim, for short) is simulation of several aspects of the phenomena in a study. It includes simulation with multiresolution models, simulation with multiaspect models, and simulation with multistage models. Multisimulation is a simulation where at decision points, simulation updates may include decisions on the branching of simulation studies as well as selection of models/submodels and associated parameters and experimental conditions to be used in subsequent stages of simulations. In multisimulation, at the interruption of the simulation, one can induce new model(s) based on the assessment of the situation so far. This very realistic possibility is not explored yet. Staging considers branching to other simulation studies in response to scenario or phase change during experimentation. Multisimulation is a proper technology to augment the mental simulation component of the decision making lifecycle shown in Figure 6. *Simulation with multimodels* allows computational experimentation with several aspects of reality; however, each aspect and the transition from one aspect to another one are considered separately. (In special cases, multimodels can be metamorphic models or evolutionary models). *Simulation with multiaspect models* (or multiaspect simulation) allows computational experimentation with more than one aspect of reality simultaneously. This type of multisimulation is a novel way to perceive and experiment with several aspects of reality as well as exploring conditions affecting transitions. While exploring the transitions, one can also analyze the effects of encouraging and hindering transition conditions. *Simulation with multistage models* allows branching of a simulation study into several simulation studies; each branch allowing to experiment with a new model under similar or novel scenarios.

5. Case Study: Symbiotic Multisimulation for Near Real-Time Decision Support in UAV Mission Analysis

In this section, we discuss a case study that involves the development of a DSS, which unifies the situation awareness and exploratory multisimulation concepts to study decision making in a multi UAV mission coordinated by a human operator. In this study, a multi-resolution coordinated mission for Unmanned Air Vehicles (UAV, which are airplanes that are flying without a human pilot on board) is being considered. The C4I system is represented with yet another simulation developed in Matlab/Simulink environment. The model is called MUAV, which is a collaborative UAV testbed (Niland 2006). The agent-augmented multisimulation based decision-making scenario examined in this scenario involves an operator that interacts with both the MUAV software that represents the C4I system and the DSS.

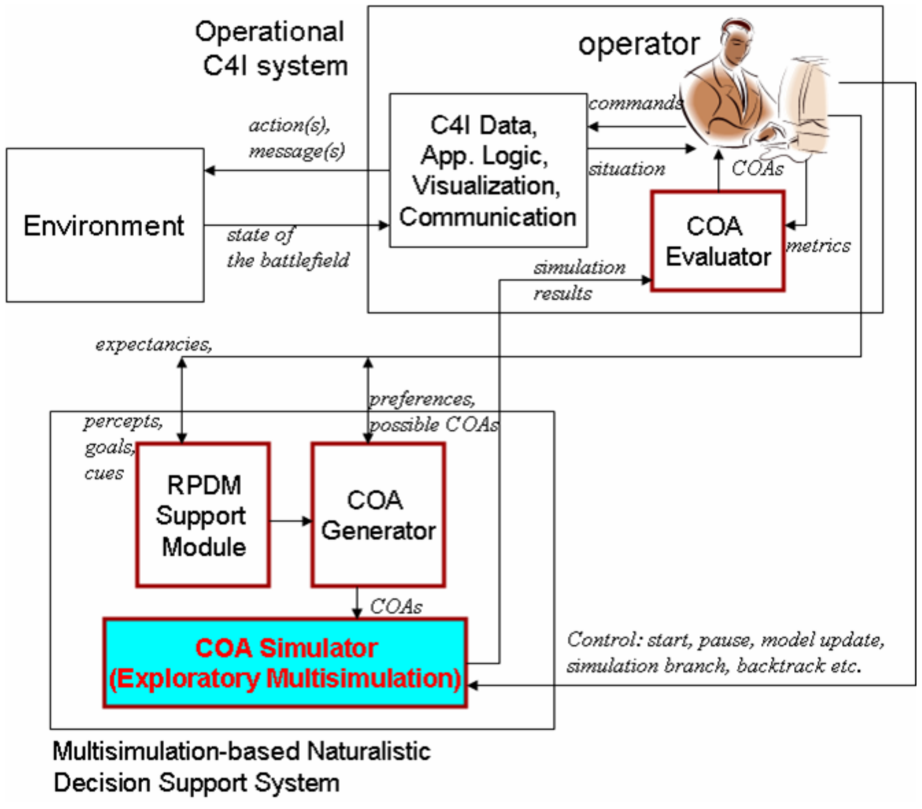

Operational C4I system
operator
C4I Data, App. Logic, Visualization, Communication
commands
situation
COAs
action(s), message(s)
Environment
state of the battlefield
COA Evaluator
metrics
simulation results
expectancies,
preferences, possible COAs
percepts, goals, cues
RPDM Support Module
COA Generator
COAs
COA Simulator (Exploratory Multisimulation)
Control: start, pause, model update, simulation branch, backtrack etc.
Multisimulation-based Naturalistic Decision Support System

Figure 10 - Architecture

Figure 10 illustrates the components of the architecture of the prototype. Currently, the subsystems of the case study are integrated and simple end-to-end scenarios can be tested. Yet, the functionality needs to be improved. The visualization components require further development to help us instill confidence in the cross resolution consistency across multiple levels of resolutions. Figure 11 illustrates the components of the case study. We are able to use MUAV as an experimental frame that drives the multiresolution simulation framework that constitutes the tactical and high resolution simulations. Tactical federate is the component with which the decision maker interacts with to use multisimulation in decision making. The interaction between MUAV and tactical federate is handled by a separate component that clusters the targets to identify battalions and establish teams based on an agent-based contract net protocol. The clustering and team formation mechanisms are explained later. Note that, the clustering and team formation algorithms are handled by intelligent agents that augment the decision making process of the operator.

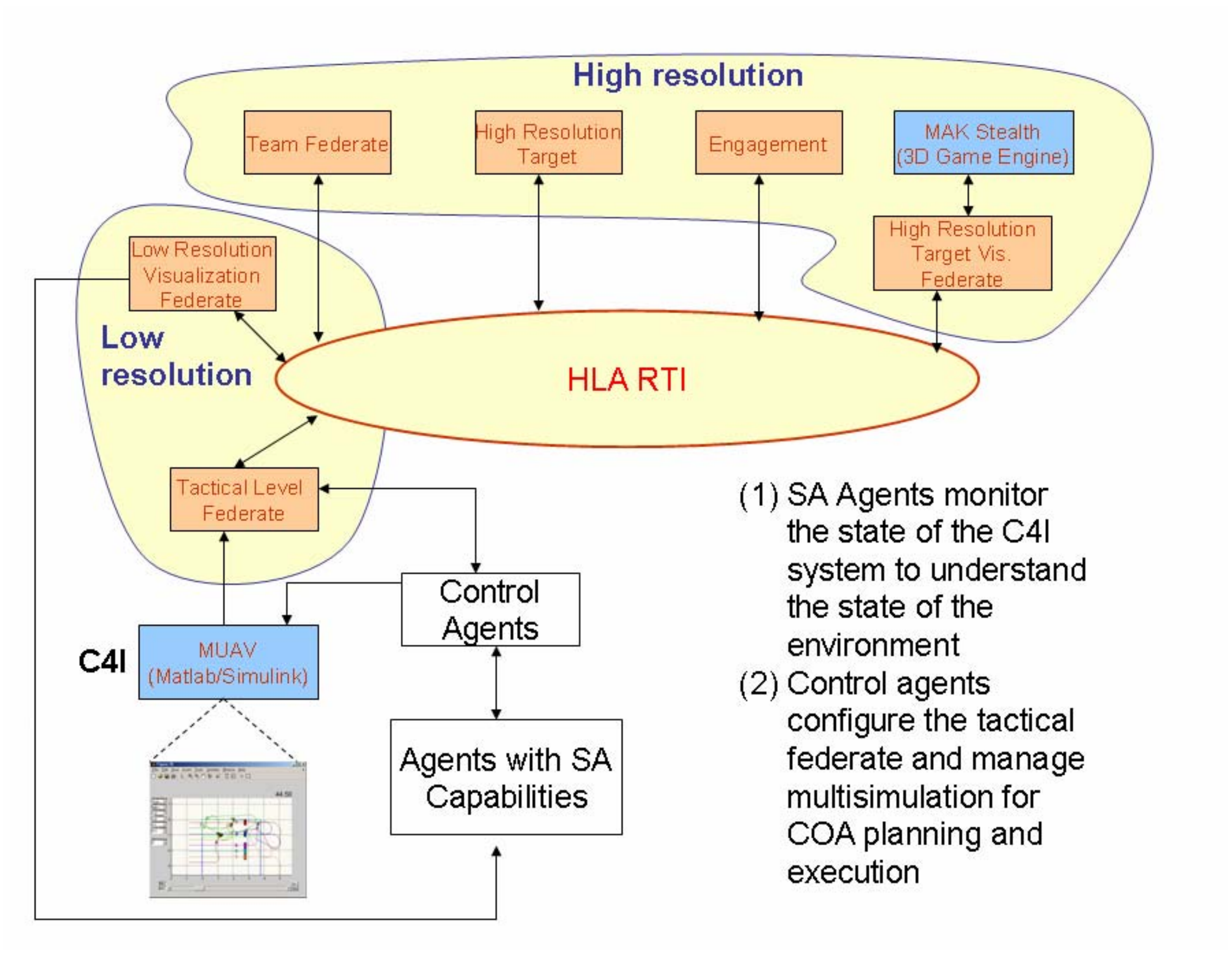

High resolution
Team Federate
High Resolution Target
Engagement
MAK Stealth (3D Game Engine)
High Resolution Target Vis. Federate
Low Resolution Visualization Federate
Low resolution
HLA RTI
Tactical Level Federate
Control Agents
C4I
MUAV (Matlab/Simulink)
Agents with SA Capabilities
(1) SA Agents monitor the state of the C4I system to understand the state of the environment
(2) Control agents configure the tactical federate and manage multisimulation for COA planning and execution

Figure 11 - A High-level View of the Components

The tactical federate uses the information passed from the MUAV to instantiate a team along with a strategy. The strategy is identified by using a Bayesian network that constitutes the anticipation element of the decision making lifecycle supported by intelligent agents. The team, high-resolution, and engagement federates, along with MAK Stealth 3D Game Engine constitute the high resolution simulation .

5.1 Situation Awareness

Situation awareness (SA) is defined as the perception of elements in a particular environment within time and space, the comprehension of their meaning and the projection of their status in the near future. Situation awareness, as depicted here, provides a set of mechanisms that enable attention to cues in the environment, and expectancies regarding future states. In realistic settings, establishing an ongoing awareness and understanding of important situation components pose the major task of the decision maker. Therefore, situation awareness is the primary basis of the decision making process in experience-based decision making process (i.e., Naturalistic Decision Making).

The understanding, diagnosis, COA generation processes are based on the following phases, which are discussed next.

- Target clustering
- Situation model generation for each UAV – target graph
- Consensus model generation for collective understanding of the situation
- Expectancy and COA generation

5.1.1 Clustering

Clustering the targets to develop a perception and consensus model constitutes the understanding component of the case study, whereas the use of a Bayesian Net enables generation of expectancies for anticipation and generation of a COA.

Algorithm:

Given a set of UN UAVs to be clustered, and a TN*TN distance (distance between each pair of targets) matrix, the basic process of hierarchy is this:

1. Start by assigning each UAV to a cluster, so that if we have UN UAVs, we now have N clusters, each containing just one UAV. Let the distances between the clusters the same as the distances between the UAVs they contain.

2. Find the closest pair of clusters and merge them into a single cluster, so that now we have one cluster less.

3. Compute distances between the new cluster and each of the old clusters.
Use Average Linkage to calculate the mean distance between clusters.

$$D(r, s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} dist(x_{ri,} x_{sj})$$

4. Repeat steps 2 and 3 until the size of one cluster is more than CS (Maximum cluster size) or the minimum distance between clusters goes above CT (distance threshold).

Metrics

1. Metrics for Undirected Graphs

   We count the discrepancies in the edges of two graphs. See the following equation:

   $$d(G_1, G_2) = \frac{1}{2} tr[(A_1 - A_2)^2]$$

   tr operator simply sums the diagonal entries in the matrix. So we can also use the following equation.

   $$d(G_1, G_2) = \frac{1}{2} \sum_i \sum_j | a_{ij}^1 - a_{ij}^2 |$$

2. Metrics for consensus model

   We compute the eccentricities of all the nodes (a node here represents a UAV graph) in the meta-level graph and select those nodes, for which the eccentricity value is minimum. And those nodes are consensus models.

Pseudo code for cluster matching:

UN: maximum number of UAVs in a team

TN: maximum number of targets (we regard it as the max number of clusters in a graph)

UAV graphs: u[0], u[0], … , u[UN-1]

Initialization:

```
For i = 0, 1, 2, … , UN-1
   For j = 0, 1, 2, … , TN-1
         u[i]. cluster[j].flag = -1
```

Find mappings:

```
For i = 0, 1, 2, … , UN-2
   For ( k = 0; k < TN && u[i].cluster[k] exists && u[i].cluster[k].flag = = -1; k++)
     {
       //It guarantees that no two flags in one graph are the same
       u[i].cluster[k].flag = sum(number of clusters in u[0], …, u[i-1]) + k;
       For j = i + 1, … , UN-1
           For (l = 0; l < TN && u[j].cluster[l] exists && u[j].cluster[l].flag = =-1; l++)
               If ( 80% of the targets in u[i].cluster[k] and in u[j].cluster[l] are the same)
                   u[j].cluster[l].flag =  u[i].cluster[k].flag
     }
```

```
//Assign suitable flags to clusters in the last graph
For ( i = 0; i < TN && u[UN-1].cluster[i] exists && u[UN-1].cluster[i].flag = = -1)
```

```
        u[UN-1].cluster[i].flag = sum(number of clusters in u[0], …, u[UN-2]) + i;
```

Rearrange the cluster array in each graph according to the flags:

```
Initialize the tmp[] array to 0 ( Each element in tmp[] is a cluster.)
For ( i = 0; i < UN; i++ )
   {
      For ( j = 0; j < TN && u[i].cluster[j] exists; j++)
        If ( u[i].cluster[j].flag!=-1)
           tmp[u[i].cluster[j].flag] = u[i].cluster[j];
      Copy tmp[] to u[i].cluster[]
   }
```

Determine neighbor clusters for each cluster in a UAV graph

```
For (i = 0 ; i < UN ; i++ )
    for( j = 0; j < TN && u[i].cluster[j].cs != 0 ; j++ )
         for ( k = j+1; k < TN  && u[i].cluster[k].cs != 0; k++)
             if (distance(u[i].cluster[j], u[i].cluster[k]) < THRES)
                 {
                  set u[i].cluster[j] as u[i].cluster[k]'s neighbor
                  set u[i].cluster[k] as u[i].cluster[j]'s neighbor
                 }
```

5.1.2 Target Graph and Consensus Model Generation for Perception

A target graph depicts the entities and their communications within a cluster. More specifically, each node in the graph represents a target, while an edge denotes the existence of a communication link between pair of targets. If the distance between two nodes is less than the pre-specified communication range, then an edge is inserted between the nodes.

Perception - The first step in achieving SA is to perceive the status, attributes, and dynamics of relevant elements in the environment. For instance, a pilot needs to perceive important elements such as other aircraft, mountains, or warning lights along with their relevant characteristics. During this task, two levels of perception capabilities will be developed. The agent based perception level then processes the raw information to derive a *target* (*situation*) *graph* that captures the enemy units and their topological layout. This helps us view enemy and friendly units as a network, so that we can apply *network analysis* methods to make inferences about a situation. Figure 12 depicts one target graph per UAV along with a meta-level graph that is used for consensus situation model generation as a product of the understanding process.

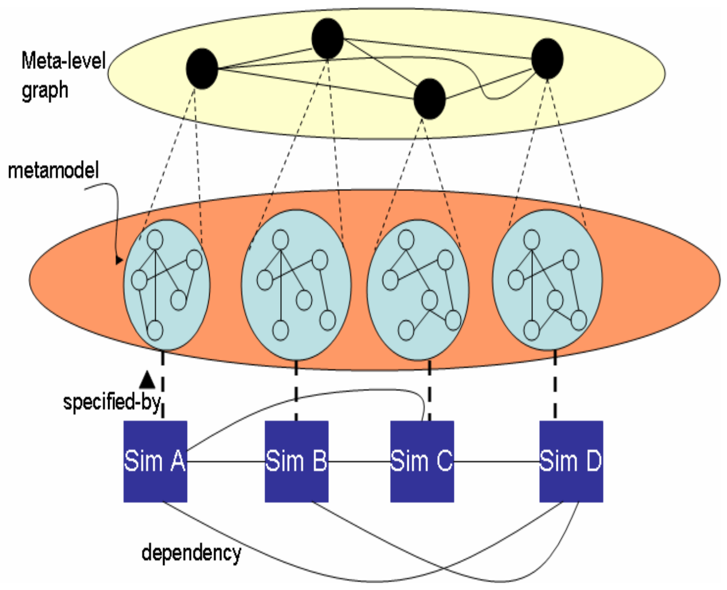


Figure 12 - Target Graphs and Consensus Model Generation

Understanding – Holistic Team Situation Model Generation: Teams perform cognitive tasks by detecting and recognizing pertinent cues, make decisions, solve problems, remember relevant information, plan, and acquire knowledge. Our position with respect to team cognition is that team situation awareness is more than the sum of the cognition of individual team members. Instead, team cognition emerges from the interplay of the individual cognitions and team process behaviors. Figure 14 presents the components of the model that is used to derive team situation model. A critical distinction that is depicted in Figure 13 is among different levels of team situation awareness. Collective perception, understanding, and anticipation portray the aggregation of individual knowledge by creating a meta-graph. Recall that individual perception and understanding involve the derivation of target graphs. The meta-level graph is defined in terms of the graphs that denote target graphs of the individual perceptions (understanding). Let G be the set of graphs and $R = R(G) = G \times G$ be a relation over the set of graphs (e.g., m_1, m_2). A dependency $(m_1, m_2) \in R$, if there exists a concrete dependency between situation s_1 perceived (understood) by a team member and situation s_2 by a team member 2, such that s_1 is specified by m_1 and s_2 is specified by m_2. The meta-graph is then defined as $V(M) = G$ and $E(M) = R(G)$.

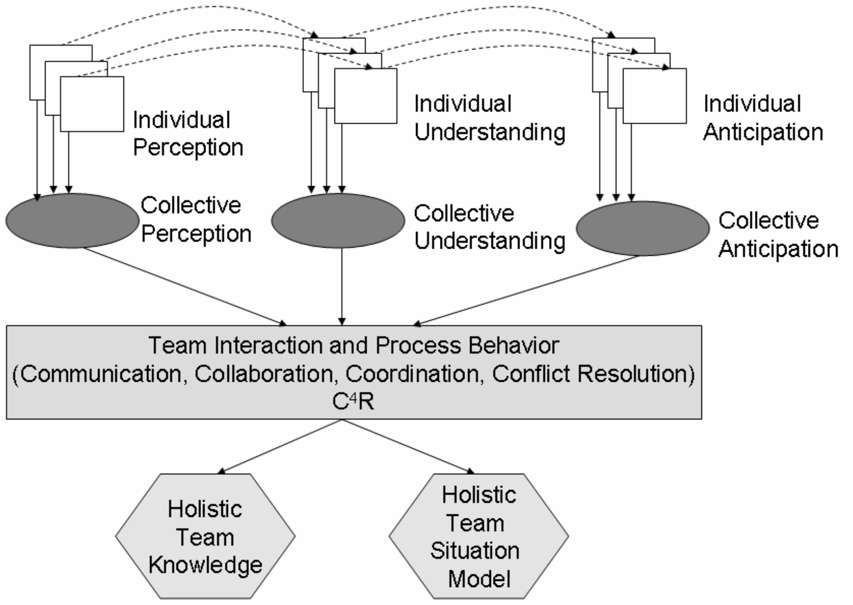


Figure 13 - Collective and Holistic Team Situation Awareness

Given that the purpose of the meta-level graph is to identify an agreement (consensus) model, the vertex set, $V(M)$, contains just those target graphs in the given set, and each graph (i.e., node in the meta-level graph) contains exactly the same edges as the original target graph. The next step in defining the meta-level graph is to identify the edges in terms of relations on the set of target graphs. A relation between two target graphs is predicated on the existence of a dependency between corresponding situations. If two situations are related, then a relation is inserted between corresponding target graph nodes. The meta-level graph is then processed to determine and assign weights on each individual relation by using graph distance metrics. These metrics measure the extent of similarity in the conceptual views of situations. The metrics can be used in conjunction with a distance threshold value so that relations (i.e., edges in the meta-level graph) can be dropped if the distance between the corresponding nodes is over the designated threshold.

The meta-level graph that depicts the meta-level perception, understanding, and anticipation frame is the basis for deriving a holistic team situation model. In our view, collective situation awareness is distinct from holistic situation awareness that emerge as a result of team processes that guide team action. We are using agent-mediated protocol that generates a consensus model among the target graphs. Our position is that it is possible to identify a consensus model from a collection of graphs by finding an element that is central in some sense to a given group of graphs.

- ***Centrality*** aims to minimize the worst case distance between the graph that includes the central nodes, which have elements with minimum eccentricity among the metamodels, and individual metamodels. Assuming that each model is represented as a node in a graph, eccentricity of a node in a graph is defined as the distance to a node farthest from *v*. Each central node can be characterized as a center, and central nodes prevent high-level dissent or discrepancy of any given node from the collection of nodes.

- Alternatively, one can try to maximize the agreement between the individual metamodels and the summary graph that is representative of the metamodel of the synthesized simulation. In other words, minimizing the average distance between the given set of metamodels and the derived conceptual model of the synthesized model (consensus metamodel), the ***median*** metric suggests a consensus model that improves the overall agreement of the collection.

Graph Similarity Metrics for Computing Centrality and Median

Graph metrics are functions that compute distances between pairs of graphs. The metamodels in our study are either undirected or directed graphs, so we focus on these two basic types of graphs. The metrics assume a common vertex set, and they are defined in terms of the adjacency matrix representation. The metrics used in this work build on prior work based on symmetric difference approach [1]. In this approach, the premise is that the difference between graphs is due to existence or absence of links between nodes. These metrics can be extended in the future to take into account the prominence of nodes as well as multiplicity (cardinality) of associations between concepts. Before presenting the metrics that measure the distance between graphs, we first specify the basic constructs and formal definitions over which the metrics are declared. Given a graph G with vertex set $V = [v_1, v_2, ..., v_n]$, the adjacency matrix A is a $n \times n$ matrix, where $a_{ij} = 1 \text{ if } (v_i, v_j) \in E(G)$.

Metrics for Undirected Graphs

In prior work presented by Banks and Carley, a symmetric difference concept is used to estimate a central graph. The strategy used in this approach is to count the discrepancies in the edges of two graphs. Its functional form for undirected graphs is given by the following equation [1]

$$d(G_1, G_2) = \frac{1}{2} tr[(A_1 - A_2)^2], \quad (1)$$

where the *tr* operator simply sums the diagonal entries in the matrix. Langfield-Smith and Wirth [11] report an alternative formulation that provides the same results. The following equation is adopted from their study.

$$d(G_1, G_2) = \frac{1}{2} \sum_i \sum_j | a_{ij}^1 - a_{ij}^2 | \quad (2)$$

Metrics for Directed Graphs

The symmetric difference metric has also been applied to directed graphs. Therefore, we will also be utilized in our approach to measure the extent of dissimilarity between metamodels. When applied to directed graphs, the metric takes slightly different algebraic form. This modified function is given by [1] as follows:

$$d(G_1, G_2) = tr[(A_1 - A_2)^T (A_1 - A_2)] \quad (3)$$

Similar to the metric for undirected graphs, the above formulation can be equivalently expressed by the following equation [11].

$$d(G_1, G_2) = \sum_i \sum_j | a_{ij}^1 - a_{ij}^2 | \quad (4)$$

Consensus Models

Finding central elements that maximize conceptual agreement (alignment) for a given collection of metamodels would provide means to measure if the synthesized model fits into its new context. The *center* and the *median* of the meta-level graph are two alternatives for computing the consensus model for the collection. The consensus model can then be compared to the application domain conceptual model to determine if the synthesized model is sufficiently close for use in this new context. If the similarity is below a particular threshold, then reused simulations can be updated (tuned) in conjunction with their metamodels until the threshold value is achieved. The threshold depends on the application domain and the extent of credibility and correspondence needed from the synthesized model. Alternatively, one can continue searching for new simulation models, the metamodels of which bring the level of agreement between the specification of the synthesized model and the application conceptual model is sufficient for the purpose of the study. There are a number of metrics for identifying central elements reported in the graph theory literature. Two of these are particularly relevant to the study presented here. The center measure that is based on the eccentricity metric enables finding those nodes in the meta-level graph that prevent high-level dissent or discrepancy of any given node from the consensus metamodel depicted by central node. On the other hand, the median metric minimizes the average distance from the metamodels of individual simulation models to the agreement model depicted by the median node. In other terms, the use of the median nodes as agreement models maximize the agreement between the individual metamodels and the summary graph that is representative of the synthesized simulation.

The Center

To determine the center nodes, one has to compute the eccentricities of all the nodes in the meta-level graph and select those nodes for which the eccentricity value is minimum. Given a meta-level graph, there are either weights associate with edges or both weights and directions. The weights depict the distance between two metamodels that is computed using the distance metrics. Given a connected graph G, let v be a node of G. The eccentricity of v is defined as

$$e(v) = \{\max d(u, v) : u \in V\}$$

The radius $rad(G)$ denotes the minimum eccentricity of the nodes. More specifically, v is central node if $e(v) = rad(G)$. The first step in identifying the central nodes is to compute distance metrics for each node in the meta-level graph. Note that using the direct dependencies between the metamodels one can compute the distance metrics, assign weights to edges, and specify them in the adjacency matrix representation of the meta-level graph.

The indirect dependencies and associated weights need to be derived by producing the distance matrix in terms of the consecutive powers of the adjacency matrix. The second step is to identify for each node v the node which has the highest disagreement (dissimilarity). The central nodes are then defined as the ones with minimum eccentricity. As such, the central nodes minimize the discrepancy between the rest of the metamodels and the agreement model represented by the central node. It is important to recognize that there can be multiple nodes with the same minimum eccentricity, and each one of these nodes is an element of the set of central nodes. Figure 8 presents the algorithm used to realize the first step of the method defined above. Using the derived distance matrix, we can identify the eccentricities of a specific node v using equation 5. Next, we need to locate the nodes with minimum eccentricity.

$$C(G) = \{u \in V(G) \mid e(u) = \min_{x \in V(G)} [\underset{w \in V(G)}{\overset{x}{E}}]\}\text{, where}$$

$$\underset{w \in V(G)}{\overset{x}{E}} = \max_w d_{xw}$$

The set $C(G)$ includes the central nodes with the same eccentricity value. If we let $C(G) = \{u_1, u_2, u_3, ..., u_k\}$, where each $u_i, 1 \le i \le k$ is a node that represents a metamodel depicted by a graph. The consensus model is selected by measuring the distance between each element of $C(G)$ and the provided conceptual model M, which specifies the context of the application domain. We use graph distance metrics to measure the distance between the selected central node and M. Let the distance between each node $u_i, 1 \le i \le k$ and M be depicted by $D_{i,M}$, then the central node with the minimum distance is $D_{j,M} = \min_i \{D_{i,M}\}$, and its metamodel is depicted by the node u_j. If we assume the existence of a threshold, δ_A, for the simulation application A, one has to assure that the consensus model is sufficiently relevant in terms of its conceptual model to the application domain model. This constraint can be evaluated by checking if $D_{j,M} \le \delta_A$. Otherwise, the simulations and/or the application constraints need to be tuned or new simulation models should be located for reuse until $D_{j,M} \le \delta_A$.

The Median

While the centrality measure aims to minimize the discrepancy of individual metamodels from the consensus model, median maximizes the agreement between the individual metamodels and the summary graph that is representative of the metamodel of the synthesized simulation. In other words, minimizing the average distance between the given set of metamodels and the derived conceptual model of the synthesized model (consensus metamodel), the *median* metric suggests a consensus model that improves the overall agreement of the collection.

Formally, let G be a connected graph. The $dist(v)$ of a node v in G is the sum of distances from v to each other node in G. This concept was introduced by Harary [10]. The median $Med(G)$ of a graph G is the set of nodes with minimum distance; the total distance $tdist(G)$ is the sum of all the distances. That is, $Med(G)$ minimizes the total distance between any vertices in the graph. Algorithms for the derivation of $Med(G)$ are presented in [3]. The following set theoretic specification provides a declarative formalization of $Med(G)$.

$$Med(G) = \{v \in V(G) \mid \sum_{w \in V(G)} d(v,w) = \min_{u \in V(G)} [\sum_{w \in V(G)} d(u,w)]\}$$

An important point to recognize is that there can be multiple nodes that exhibit the characteristic of a median node. In that case a strategy that is similar to the one discussed in section 4.4.1 can be used to select the node that minimizes the distance against the simulation application's conceptual model. Furthermore, the threshold value can be used to assess the fitness of the consensus model. It is also important to notice that if $C(G) \cap Med(G) \neq \Phi$, then there exist nodes that act both as central and median. Such nodes are particularly good candidates for being consensus models, as they not only minimize dissent of any given metamodel, but also maximize the agreement between the individual metamodels and the consensus model.

5.1.3 Expectancy and COA Generation for Exploration

Following the generation of the consensus model, COAs are generated based on expectancies. Expectancies and COAs are based on successful matching of the parameters and characteristics of the situation model to learned experiences that are captured by a predictive Bayesian model.

For the Strategy Decision, we use Bayesian Network to choose the more plausible and proper strategy. A Bayesian network (or a belief network) is a probabilistic graphical model that represents a set of variables and their probabilistic dependencies. We use the Microsoft Bayesian Network tools MSBNx to implement the Bayesian Network. MSBNx is a tool for doing that kind of cost-benefit reasoning for diagnosis and troubleshooting.. The MS Bayesian Network in our project is implemented as shown in Figure 7. The top nodes are Area, Betweenness, Degree, CommRang (Communication Range), TargetNumber, UAVNumber, FSEfficiency (Fringe Point Strategy Efficiency), FSEffectiveness (Fringe Point Strategy Effectiveness), DCSEfficiency (Destroy Communication Range Strategy Efficiency), DCSEffectiveness (Destroy Communication Range Strategy Effectiveness). The top nodes are the input nodes that receive the input data to initialize the Bayesian Network. The bottom nodes are DCS (Destroy Communication Range Strategy) and FS (Fringe Point Strategy) which are the strategies we need to choose for UAV team action as shown in Figure 14.

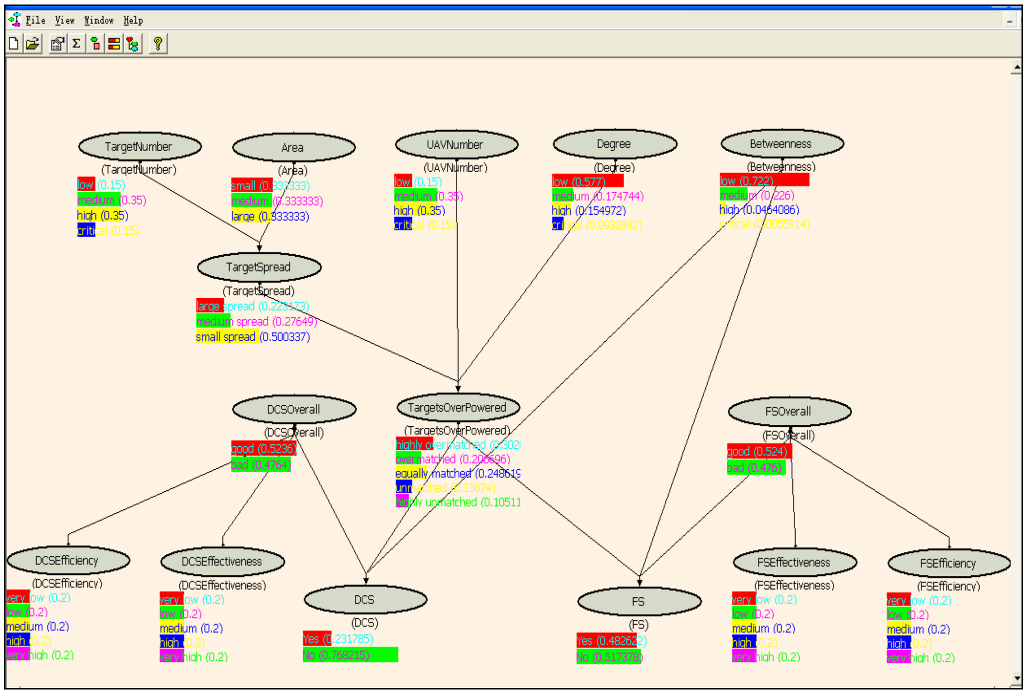
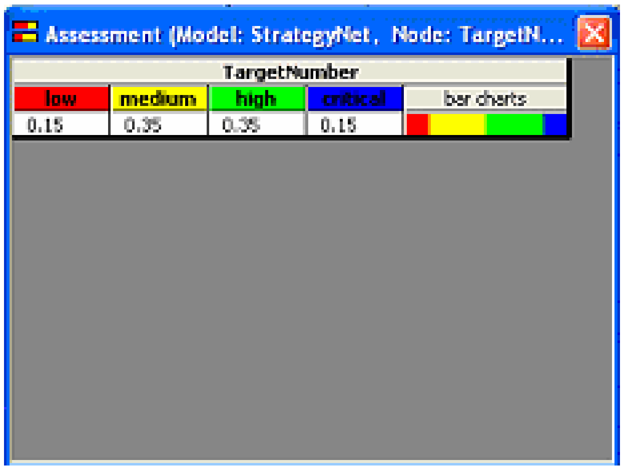


Figure 14 - Bayesian Network

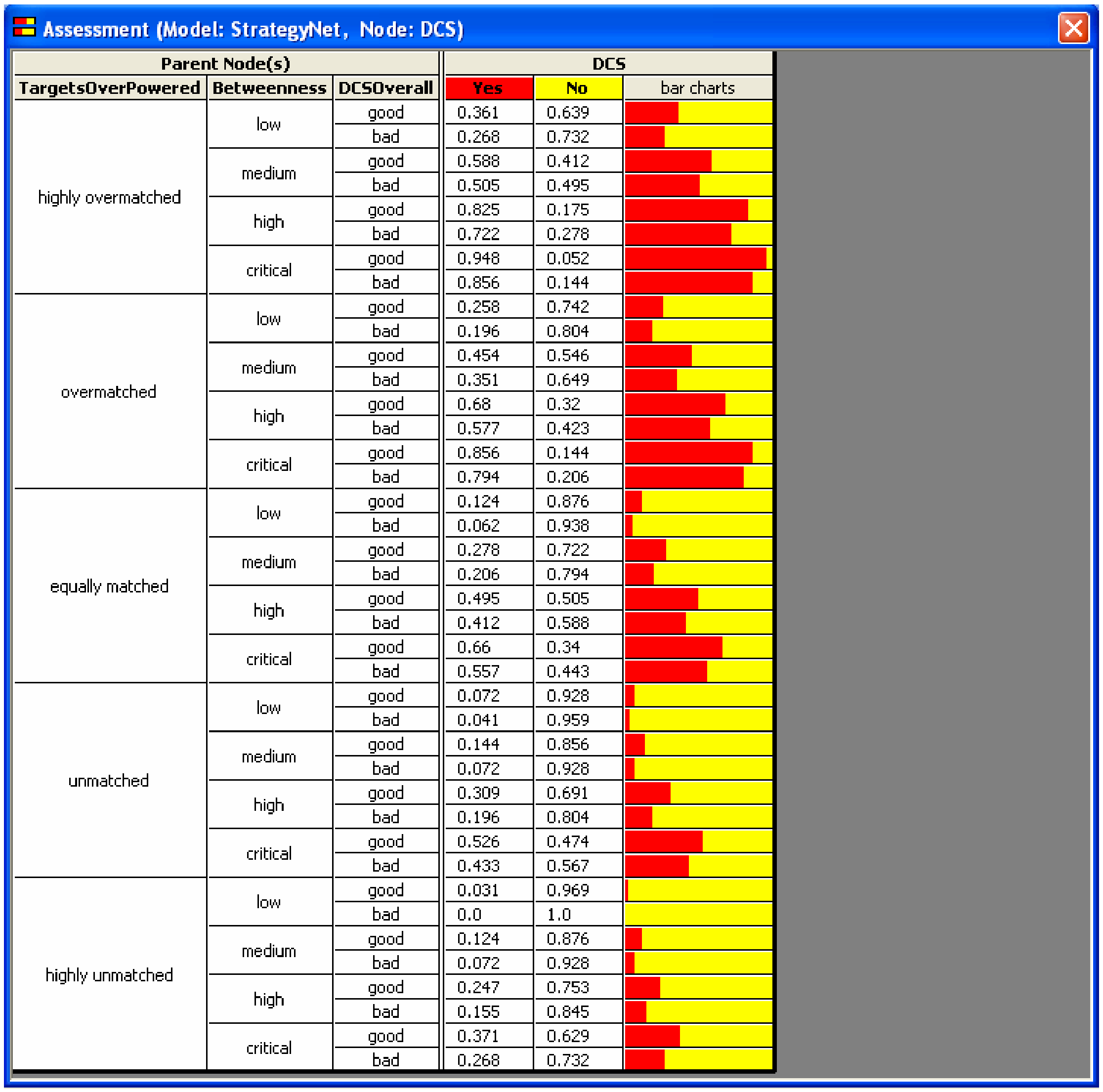
The Bayesian Network is built as follows: First, we create a diagram for the cause-effect relations. Then, we set uncertainty in the Assessment module for each node in the system, giving the probabilities for various situations. For example, we set the number of targets to four levels: low, medium, high, critical with the probability of low 0.15, medium 0.35, high 0.35 and critical 0.35 as shown in Figure 15.



Assessment (Model: StrategyNet, Node: TargetN...

TargetNumber				
low	medium	high	critical	bar charts
0.15	0.35	0.35	0.15	🟥🟨🟩🟦

Figure 15 - Probability table of “Number of Target” in various situations.



Assessment (Model: StrategyNet, Node: DCS)

Parent Node(s)			DCS		
TargetsOverPowered	Betweenness	DCSOverall	Yes	No	bar charts
highly overmatched	low	good	0.361	0.639	
		bad	0.268	0.732	
	medium	good	0.588	0.412	
		bad	0.505	0.495	
	high	good	0.825	0.175	
		bad	0.722	0.278	
	critical	good	0.948	0.052	
		bad	0.856	0.144	
overmatched	low	good	0.258	0.742	
		bad	0.196	0.804	
	medium	good	0.454	0.546	
		bad	0.351	0.649	
	high	good	0.68	0.32	
		bad	0.577	0.423	
	critical	good	0.856	0.144	
		bad	0.794	0.206	
equally matched	low	good	0.124	0.876	
		bad	0.062	0.938	
	medium	good	0.278	0.722	
		bad	0.206	0.794	
	high	good	0.495	0.505	
		bad	0.412	0.588	
	critical	good	0.66	0.34	
		bad	0.557	0.443	
unmatched	low	good	0.072	0.928	
		bad	0.041	0.959	
	medium	good	0.144	0.856	
		bad	0.072	0.928	
	high	good	0.309	0.691	
		bad	0.196	0.804	
	critical	good	0.526	0.474	
		bad	0.433	0.567	
highly unmatched	low	good	0.031	0.969	
		bad	0.0	1.0	
	medium	good	0.124	0.876	
		bad	0.072	0.928	
	high	good	0.247	0.753	
		bad	0.155	0.845	
	critical	good	0.371	0.629	
		bad	0.268	0.732	

Figure 16 - Probability table of DSC in various situations

Bayes Net: First Level

The Bayesian net takes input variables in at the top layer of the net and with these inputs it suggests a strategy for the team to use. The Bayesian net is compromised of 9 input variables: UAVNumber, TargetNumber, Area, CommRange, DCSEffectiveness, DCSEffieciency, FSEffectiveness, and FSEfficiency.

- UAVNumber is the amount of units in the team
- TargetNumber is the amount of units in the Battalion
- Area is the area in which the battalion occupies

- CommRange, also communication range, is the overall average of the battalions communication capabilities. For example, if they have a high range, the battalions can communicate over long distances.
- Speed is not built into the system or defined, however it is shown in the model
- DCSEffectiveness, which is the effectiveness for the Destroy Communication Strategy (DCS). This is a history variable provided by the performance in the Team Federate.
- DCSEfficiency, efficiency for the DCS
- FSEffectiveness, effectiveness for the Fringe Strategy (FS)
- FSEfficiency, efficiency for the FS

Effectiveness and Efficiency are values provided by the high resolution simulation. They let the Tactical Federate know how the high resolution team is performing. With these values, the tactical federate can decide to choose a new strategy over the current strategy running in the high resolution team.

Bayes Net: Second Level

The second level of the bayes net consist of TargetSpread, DCSOverall, and FSOverall

- TargetSpread measures how spread out the target is over the area
- DCSOverall combines DCSEfficiency and DCSEffectivness into one varaiable
- FSOveral combines FSEfficiency and FSEfficiency into one variable

Bayes Net: Third Level

The third level of the Bayes Net consists of degree, betweeness and closeness.

- Degree is graph theory. It tells how many connections a node has to other nodes. It helps to form an idea of a danger level. The more connections a node has, the more dangerous it may appear
- Betweenness is also a part of graph theory. In this context, we are talking about communication betweenness. Betweenness is the measure of shortest paths flowing through one node. The more shortest paths that a node has flowing through it, the higher its betweenness is. If a UAV can take out the node with the highest betweenness, the UAV can make a big dent in the communication structure of the Battalion.

Bayes Net: Fourth Level

- TargetsOverPowered is a ratio of the power of the UAVs to the power of the Battalions. It takes in considerations of degree which is a calculation of spread.

Bayes Net: Fifth Level - Strategies

The fifth level holds the strategies that can be chosen by the Tactical Federate. The two strategies that exist are the Fringe Strategy(FS) and the Destroy Communication Strategy(DCS).

- DCS, this strategy destroys the nodes with the highest amount of communication betweenness.
- FS, this strategy works from the fringe, or edge, of the battalion to the center. This is a more conservative strategy that is used when the UAVs team is small and the Battallion is large or dense.

Based on the Bayesian Network, we can calculate the probability of choosing Fringe Point Strategy or Desestroy Communication Range Strategy. We set the strategy with higher probability as the strategy with the highest priority and pass the strategy information to engagement. The engagement with team and strategy information is passed to Team Federate. When the high resolution is running, it will pass back the information to Tactical Federate. The Tactical Federate will periodically pass this information to the Filter to calculate the new Cluster and Team. After calculation, Filter sends the information to Tactical Federate. After receiving the updated information from Filter, Tactical Federate sends the updated information to LRV. Tactical Federate calculates the betweenness based on the new information and then calculates new strategy using Bayesian Network with the updated input data such as the number of UAV and target left, area, betweenness, degree, the efficiency and effectiveness for each strategy. It calculates the new probability for each strategy with the new input information. After the calculation by Bayesian Network, Tactical Federate passes the new strategy, team and cluster information for each team to Team Federate. Team Federate will adjust the team formation based on the new probability of each strategy and update the environment in high resolution based on that information.

5.1.4 Team Formation

Figure 17 illustrates the protocol for team formation to attack an aggregate target. When a team needs to attack the cluster, the first UAV that finds the target cluster becomes the Team leader. The team leader decides the number of team members based on the cluster size. The team leader then sends CFP (Call for Proposals) information to the N UAVs who has the minimal distance between it and the cluster center. The UAV receives the CFP and checks if it is willing and available to be a team member. The UAV has the certain random chance of willing to be a team member. If the UAV has the willing to be the team member, it accepts the proposal. It sends the accept information to the team leader and makes a team contract with Team leader. If the UAV rejects the proposal, it sends reject information to the team leader. If the there is not enough team members in the team, the team leader will send CFP to other UAVs based on the distance between them and the cluster center until there are N UAVs in the team. If all the other UAVs have been in a team, the team formation process stops.

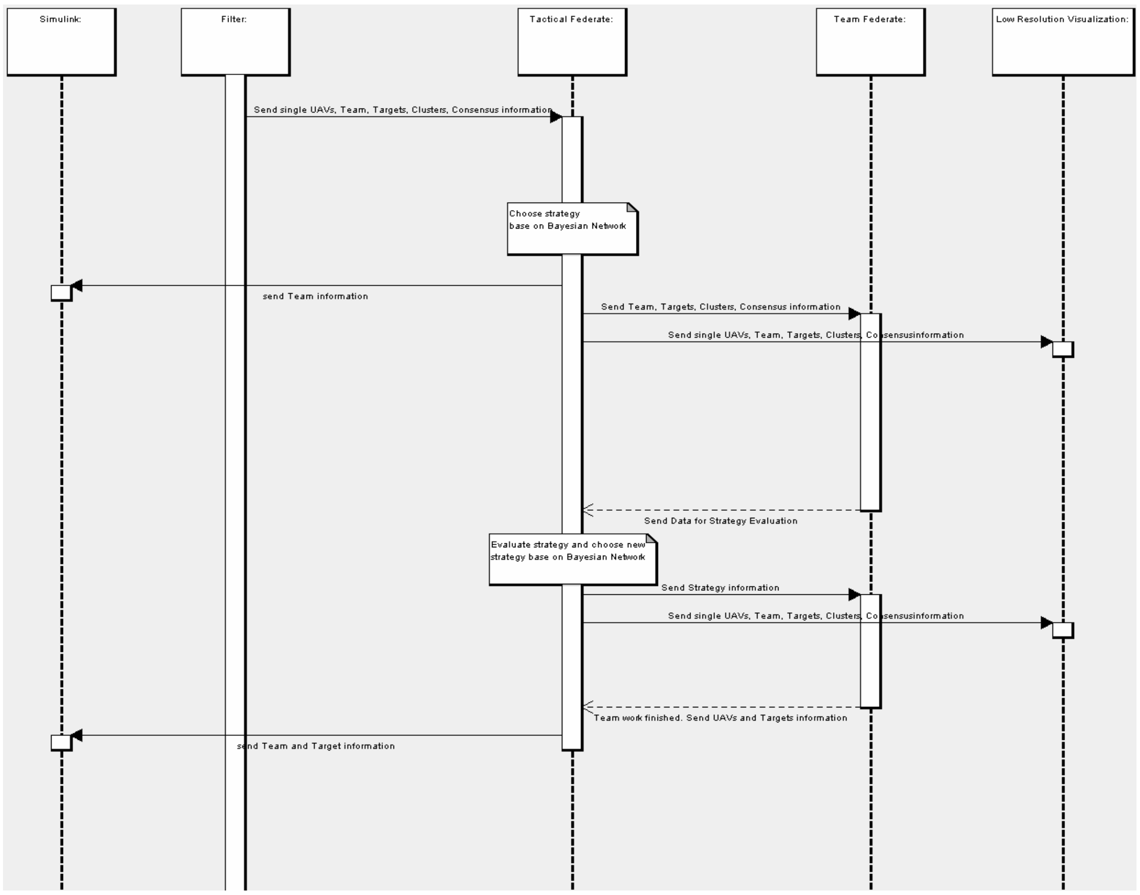

Simulink:
Filter:
Tactical Federate:
Team Federate:
Low Resolution Visualization:
Send single UAVs, Team, Targets, Clusters, Consensus information
Choose strategy base on Bayesian Network
send Team information
Send Team, Targets, Clusters, Consensus information
Send single UAVs, Team, Targets, Clusters, Consensusinformation
Send Data for Strategy Evaluation
Evaluate strategy and choose new strategy base on Bayesian Network
Send Strategy information
Send single UAVs, Team, Targets, Clusters, Consensusinformation
Team work finished. Send UAVs and Targets information
send Team and Target information

Figure 17 - Tactical Federate Sequence Diagram

5.1.5 Screen Shots of Low Resolution Simulation Visualization

The following pictures depict the visualization of the low-resolution situation awareness simulation. Eight UAV’s represented by rectangles fly over some ground targets and analyze target cluster characteristics.

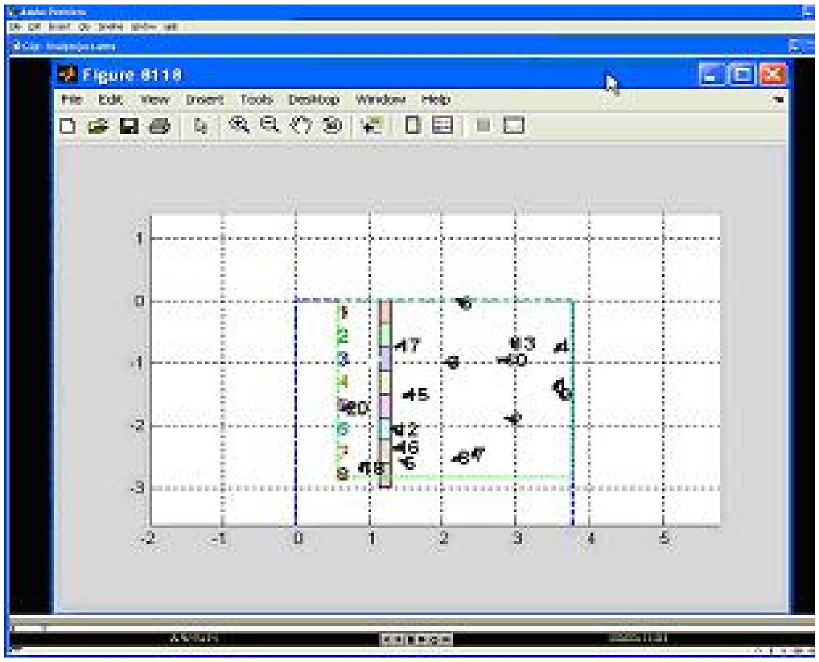
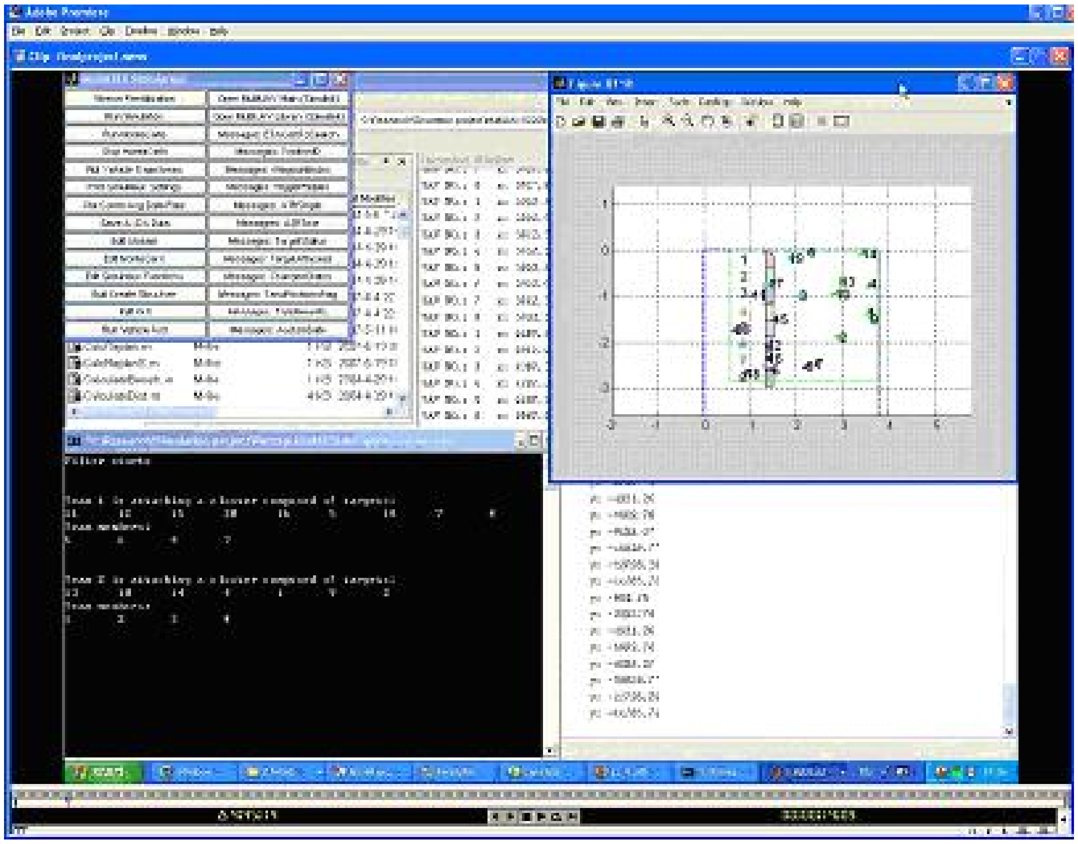


Figure 18 - Low Resolution Simulation

Once targets cluster information is determined, the UAVs form appropriate teams, as shown in the bottom left window. The appropriate COA's are also generated to engage the targets.



```
Teams starts


Team 1 is attacking a cluster composed of targets:
21      12      15      20      16      5       18      7       8
Team members:
5       6       8       7


Team 2 is attacking a cluster composed of targets:
13      10      14      4       1       9       3
Team members:
1       2       3       4
```

Figure 19 – Simulation Start-Up

Two teams of UAVs moving into position to engage the targets using COAs determined by the control agent or the operator.

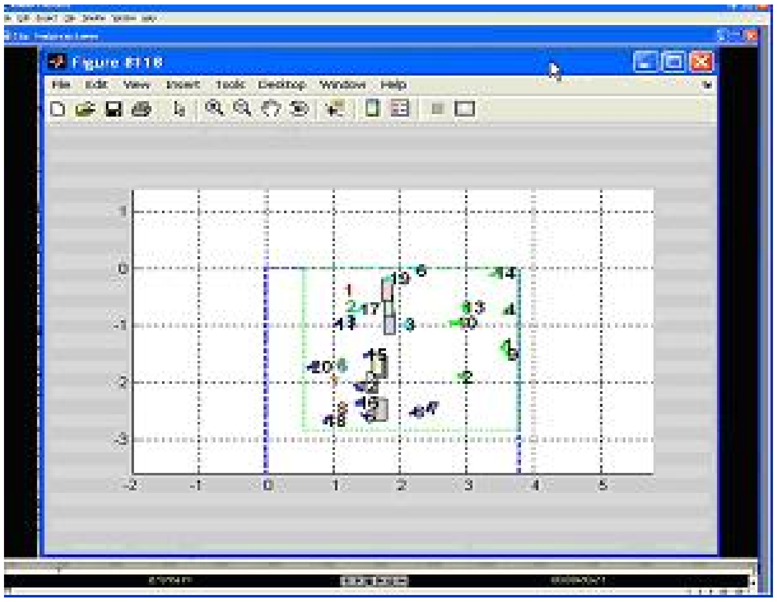
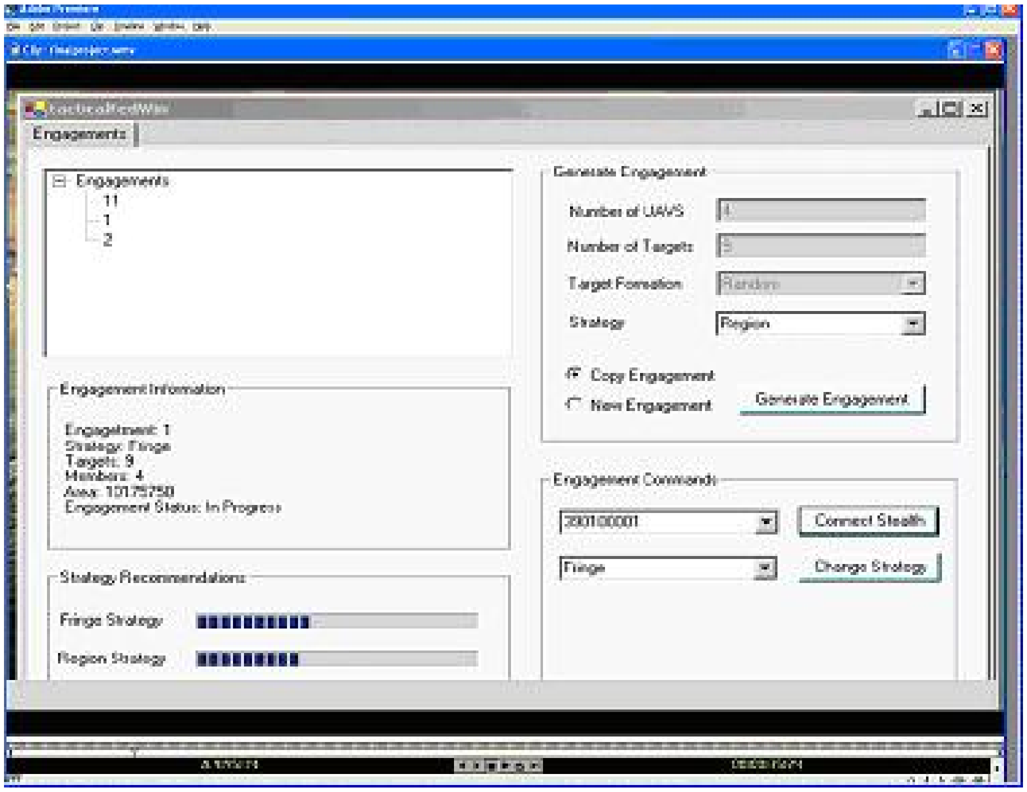


Figure 20 – Team Formation

The tactical federate GUI interface allows operators to clone multiple parallel COA's simulation using different strategies. Two different engagements of the two teams are also shown. Multiple COA's strategies of these engagements is simulated using high-level resolution multi-simulation



Engagements
- 11
- 1
- 2

Number of UAVS: 4
Number of Targets: 9
Target Formation: Random
Strategy: Region

Engagement Information

Engagement: 1
Strategy: Fringe
Targets: 9
Members: 4
Area: 10175750
Engagement Status: In Progress

300100001

Fringe

Fringe Strategy

Region Strategy

Figure 21 – COA Selection

5.2 Exploratory High-resolution Simulation of Strategies using Multisimulation

The goal of this part of the project is to promote multiresolution, multistage models or MRMSM. A Multiresolution model contains one or more models that capture a phenomenon at differing levels of resolution. A Multistage model rotates through different stages of existence. Multistage is much like a student moving from one grade to the next; the student is changing stages of his life.

In our project, we are focused primarily on two different resolutions, high and low resolution teams. At the high level, we are concerned with individual UAVs and how they cooperate with each other to destroy a Battalion of tanks. The low resolution model aggregates these UAVs and gathers aggregated information about the team status, the engagement status, and the effectiveness and efficiency of the strategy. (See Multi Resolution Design)

We promote multistage resolution by supporting alternating strategies. A team can alternate to strategy to adapt to the problem at hand. Three of the stategies that we have identified in this project are the Fringe Strategy, the Basic Strategy and the Destroy Communication Strategy. The Basic Strategy is more like a test strategy and will not be used later on. (See Strategy Design)

To help deduce which strategy the problem space demands, we make use of a Bayesian Net. It takes several inputs, such as number of UAVs or number of targets, and decides a strategy. The low resolution monitors how the team is doing with a given strategy. If the problem space changes, the low resolution team will notify the tactical federate to choose new a new strategy that meets the new needs of the problem space. (See Baysean Net Design)

Another aspect in our project is to simulate several instances of the same problem at the same time, or MultiSimulation. To recognize this goal, we need a way to separate the presentation space between each instance of the problem. This separation is hoped to be realized by using the DDM (Data Distribution Management) aspect of HLA. (See Stealth and Engagement Manager Section)

5.2.1 Tools

To help facilitate our goals, we are using the DoD standard HLA. We are specifically using MAK's HLA 1.3 implementation of the RTI. We are also using MAK VR-LINK which is an independent interface for several types of simulation standards, like DIS and HLA.

For our presentation of the High Resolution Model, we are using MAKs Stealth. The MAK Stealth is built upon VR-LINK. By using VR-LINK with our federates, we are able to easily communicate with Stealth.

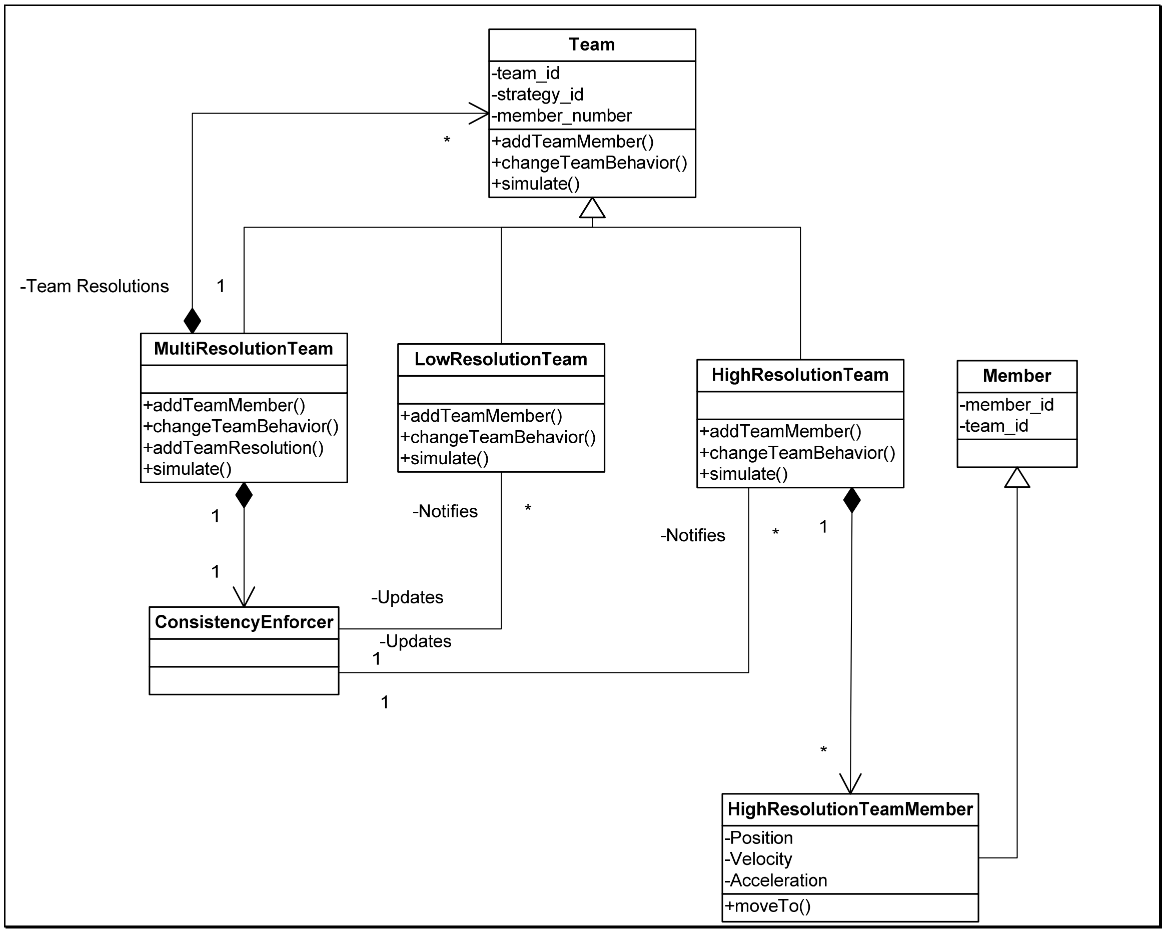


Figure 22 – UML Class Model for UAV Team Design

5.2.2 Multi Resolution Design

The overall structure of this design is the composite pattern. In this structure *Team* would be the **Component**, *LowResolutionTeam* and *HighResolutionTeam* would be **Leaf** nodes, and *MultiResolutionTeam* would be the **Composite.** A MultiResolution team is composed of a group of team resolutions. Notice how this design allows for a hierarchy; multi-resolutions within multi-resolutions. These resolutions should be thought of more as a list rather than a tree. Since multi-resolution should be viewed as list, another design maybe more appropriate. Having tree-like composition may allow logical errors into the system.

The idea is for the Multi-Resolution to contain different resolutions. In our project, our multi-resolution team includes two different resolutions, low and high. When a client uses addTeamMember () to a multi-resolution team, it will add members to all the resolutions within. Each level or resolution, will decide how to add that member. For example, when high resolution gets the call to add a member, it will create an instance of a high resolution team member, whereas a low resolution will just increase the member number counter.

The Multi-Resolution Team also contains the consitency enforcer (CE), this is meant to keep consitency between the different resolutions containted within. Each resolution within contains a link to the consitency enforcer so that it can notify the CE when a significant change has taken place. In turn, the CE will update the resolutions that care about the change. This design is much like the mediator-observer pattern.

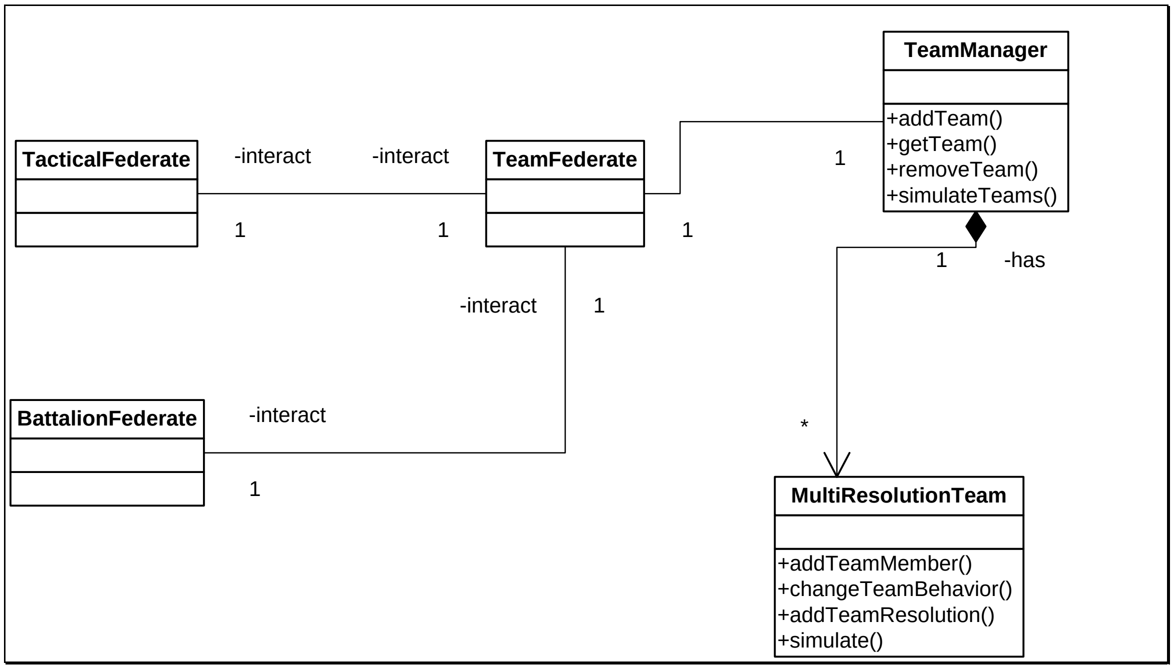


Figure 23 – Team Federate

5.2.3 Federate Class Design

The federate class design is rather simple. Each federate (written using HLA and VR-Link) is a singleton. There is only one instance, or global instance, needed in a program at a time. The Team federate receives data from the tactical federate and from the battalion federate. The team federate will then in turn interact with the team manager or any other type of Managers needed.

Strategy Class Diagram - This class design is actually the strategy design pattern. It allows for the High Resolution Team to change out its Course of Action or Behavior at runtime. As of now, only the Basic Strategy is implemented. Basic plans its strategy by choosing enemies with a first come first served policy. When Fringe is implemented, it will plan its strategy by choosing enemies on the outskirts.

This is a very simplified version of a strategy. My strategies can further be decomposed into several strategies. There could be several planning strategies, path finding strategies, assignment strategies, and different running strategies.

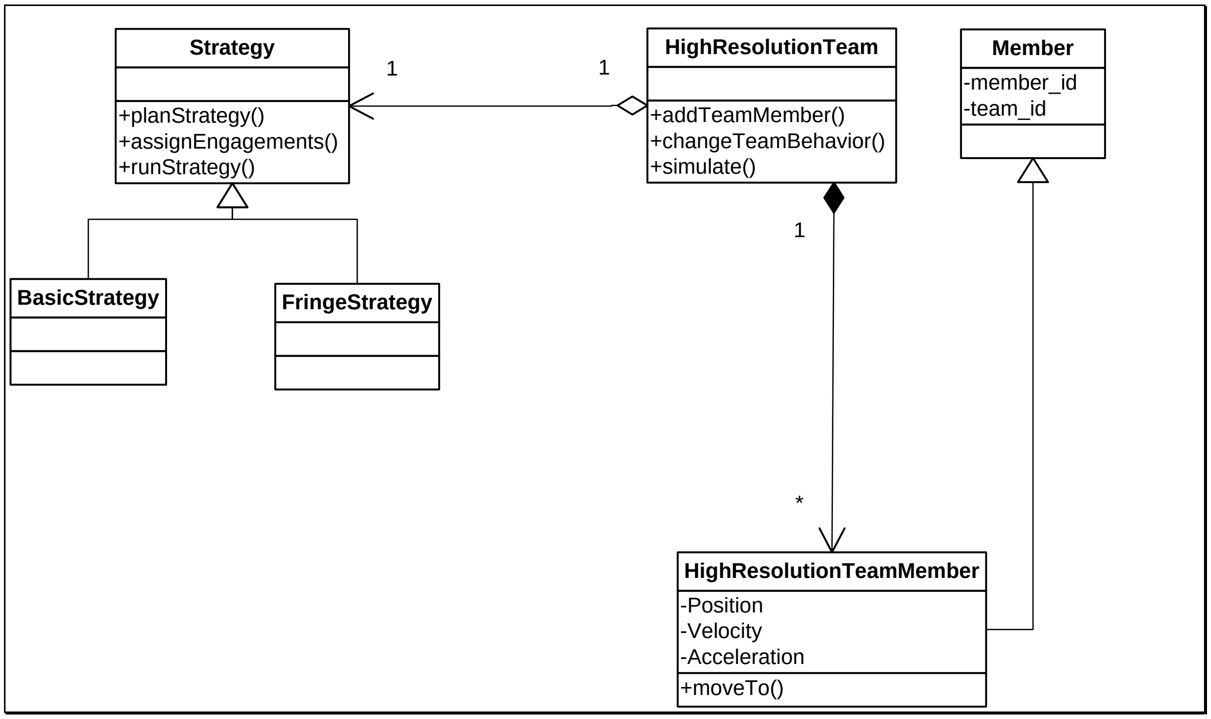


Figure 24 - Strategy Design Pattern

The two strategies that we identified in our previous research is the Fringe Strategy and the Destroy Communications Strategy:

- DCS, this strategy destroys the nodes with the highest amount of communication betweeness.
- FS, this strategy works from the fringe, or edge, of the battalion to the center. This is a more conservative strategy that is used when the UAVs team is small and the Battallion is large or dense.

Stealth Manager and Engagement Manager

The Tactical Federate contains two major components, the Stealth Manager and the Engagement Manager. These two managers keep up with all the stealth and engagement instances. In later periods, these managers will become subjects of the Observer design pattern. They represent the model of data that is to be observed by some type of UI. At this point, each manager contains a default presentation function “print”. This presentation function prints to the console a simple list of the currents engagements or stealth instances. The user can use this information to connect a stealth instance to an engagement instance.

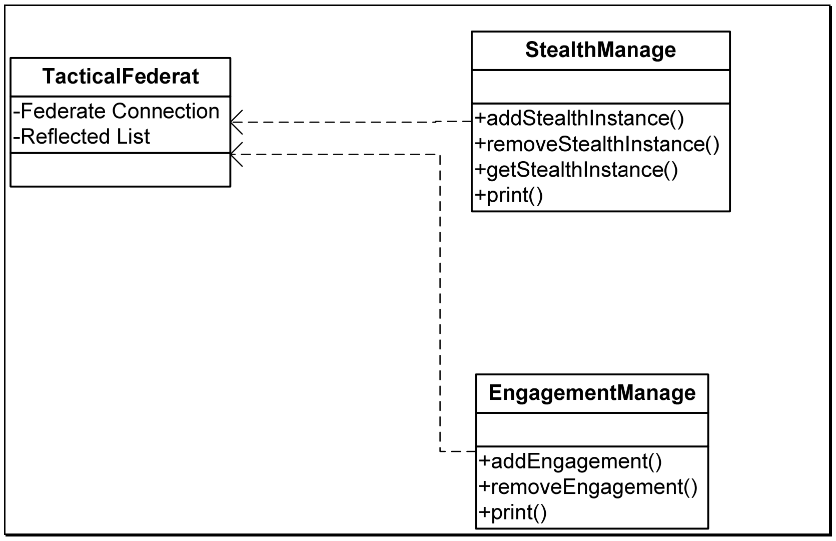


Figure 25 - Stealth Manager

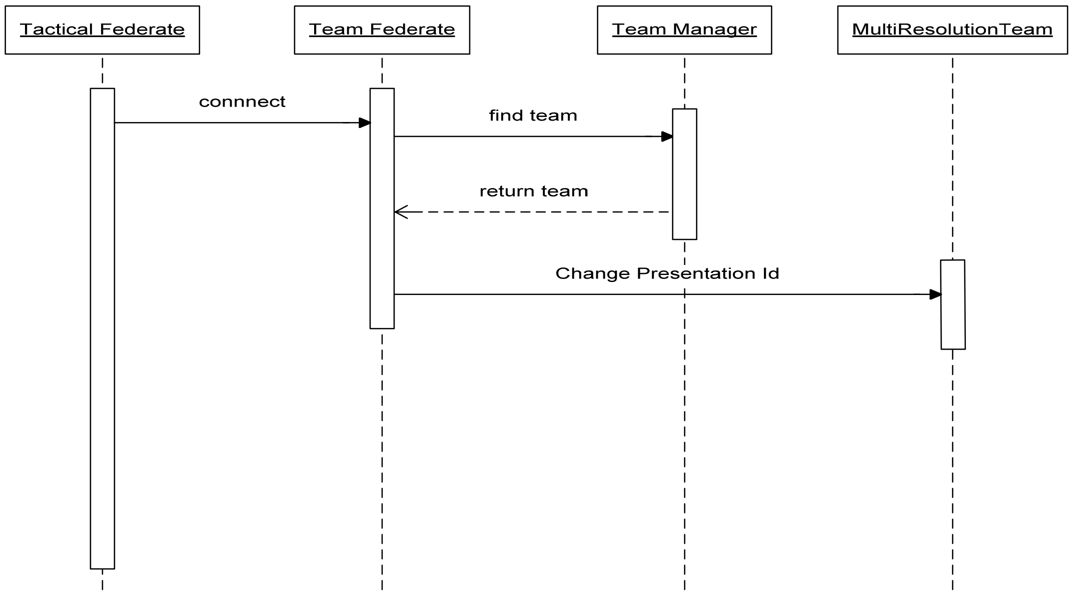


Figure 26 – Interaction Pattern for Team Formation

When a user call connects from the Tactical Federate, it finds the Engagement Instance in the Team Federate from the team manager and changes the team's presentation ID. The Team uses this presentation ID to set its Region information for the DDM in the RTI. The DDM allows the Stealth instances the ability to choose which engagement it will display.

Create Team And Add Team Member Sequence Diagram
This sample sequence diagram shows what happens when a team is added and a team member is added. A tactical federate will ask the team federate to add a team. In turn the Team Federate will give the team manager the responsibility to actually create the team.

When adding a team member, the tactical federate sends a member to the team federate. The team federate hands this information to the team manager. The team manager finds the right team to add the member to, then it asks that team to add a member. Notice how all responsibility seems to be delegated across.

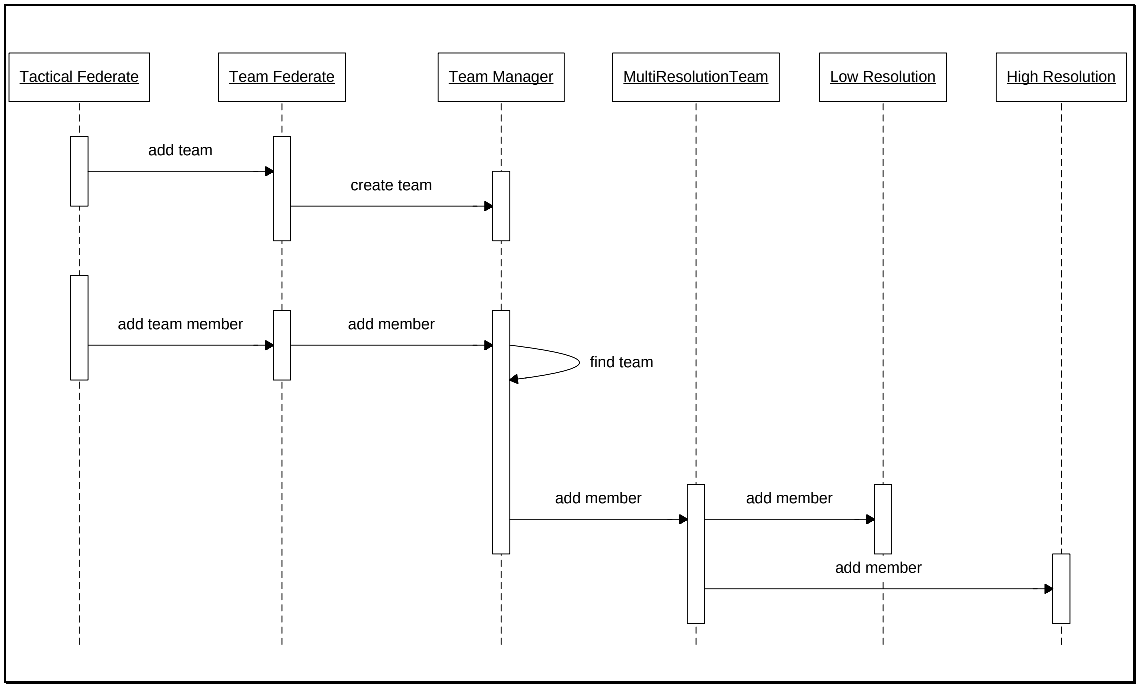


Figure 27 – Team Population

5.2.4 Screen Shots of High Resolution Simulation Visualization

The following pictures depict the visualization of the high-resolution engagement simulation. The first picture depicts the visualization of the simulation of the fringe strategy by Team 1.

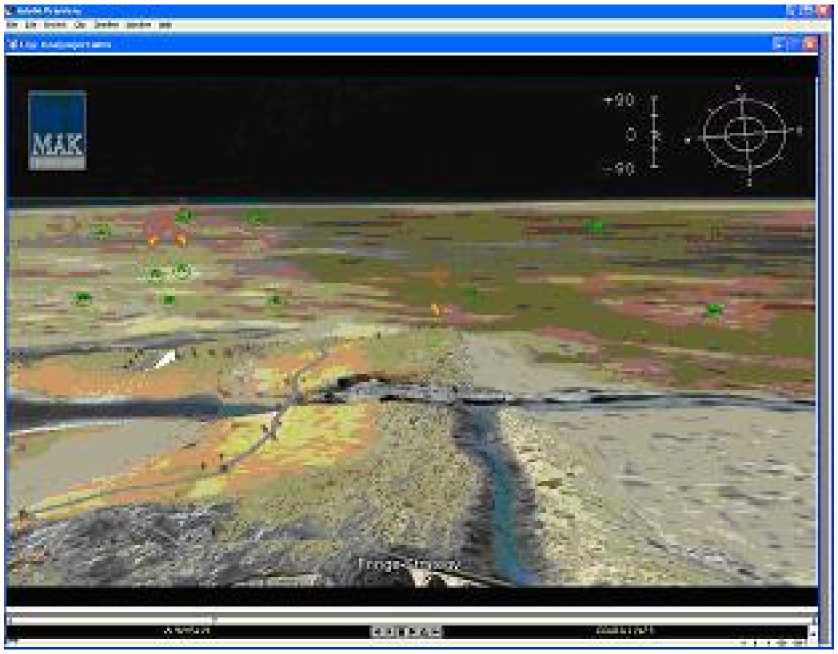


Figure 28 – Fringe Strategy

The following screen shows the visualization of the simulation of the region strategy by Team 1.

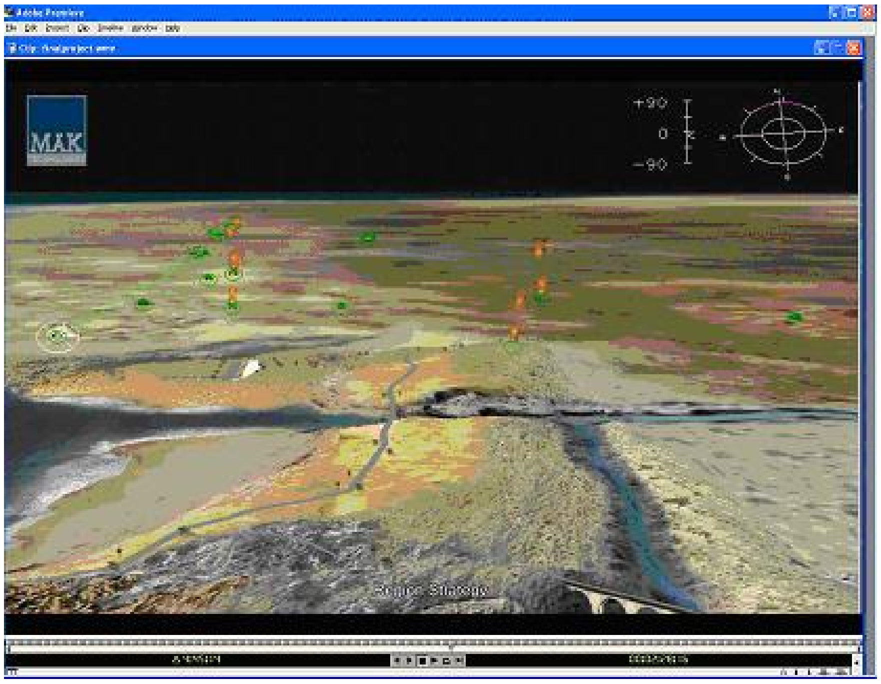


Figure 29 – Region Strategy

This screen shows the results at the end of the simulation of the region strategy by Team 1, where the number of remaining targets is shown to be 0 (all destroyed).

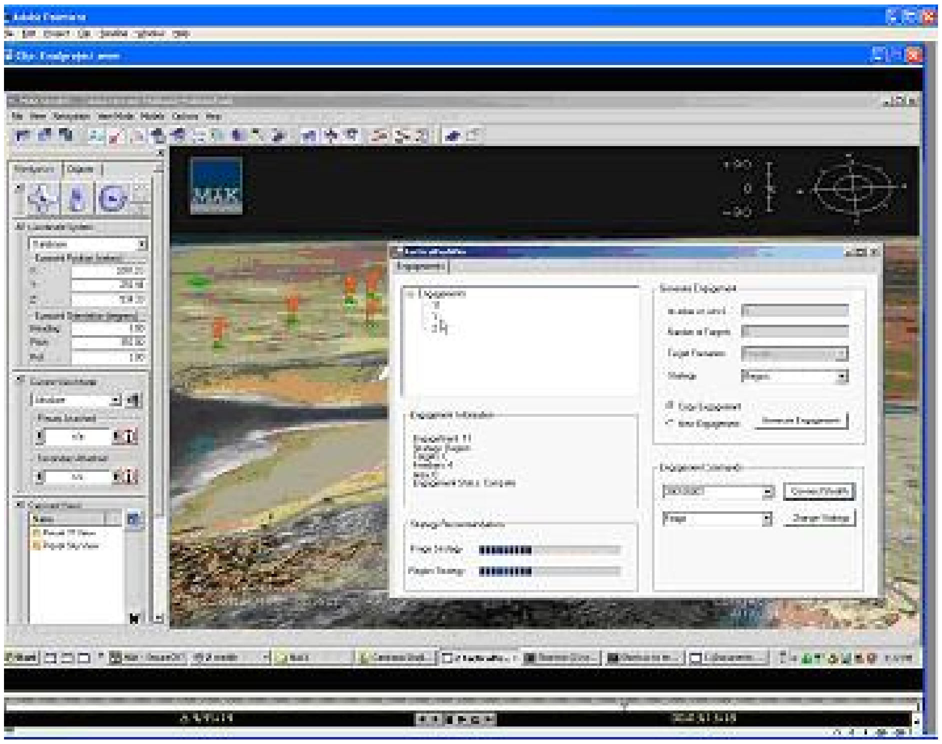


Figure 30 - Completion of Region Strategy

The following pictures depict more visualization of the high-resolution engagement simulation.

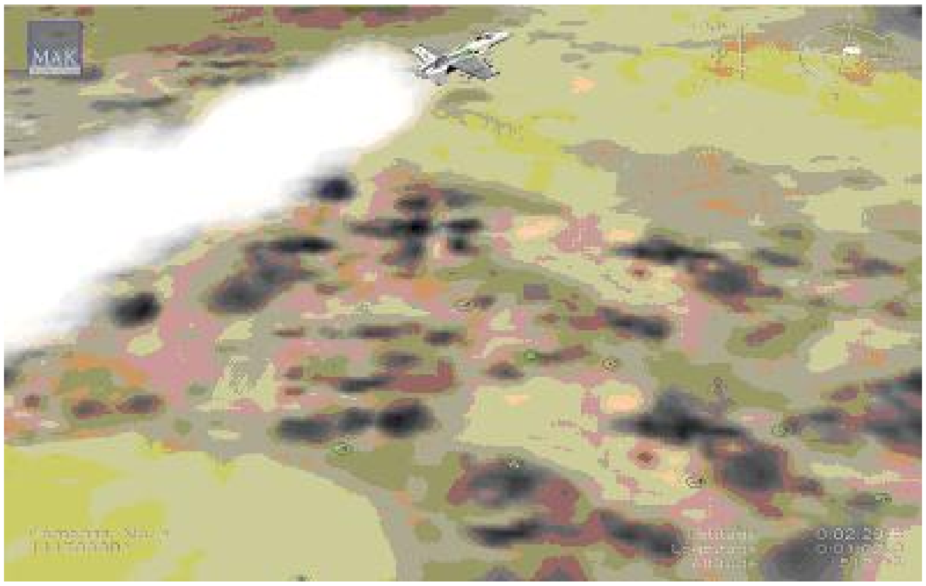


Figure 31 – UAV Attack

Figure 32 - Targets

6. Requirements and Design Principles for Next Generation Simulation-based Decision Support Systems

While the current case study does not support the features advocated in this formalism due to limitations of the HLA infrastructure, future research will realize this proposed framework.

The proposed framework is predicated on the requirements and design principles for Multiresolution Multistage Multimodels (MRMSM)

.

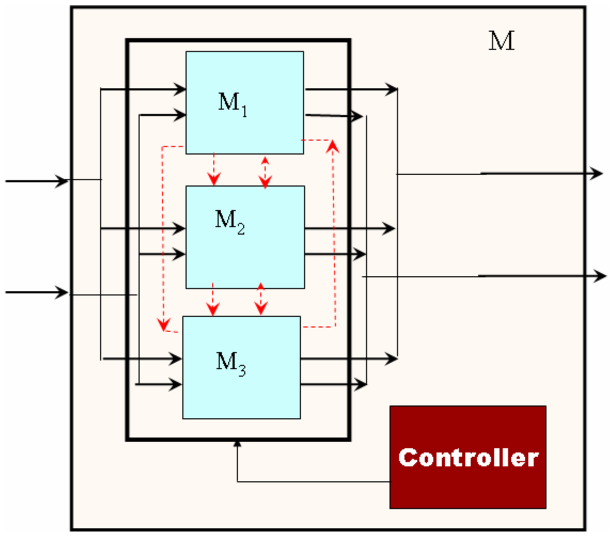


Figure 33 - Multiresolution Model

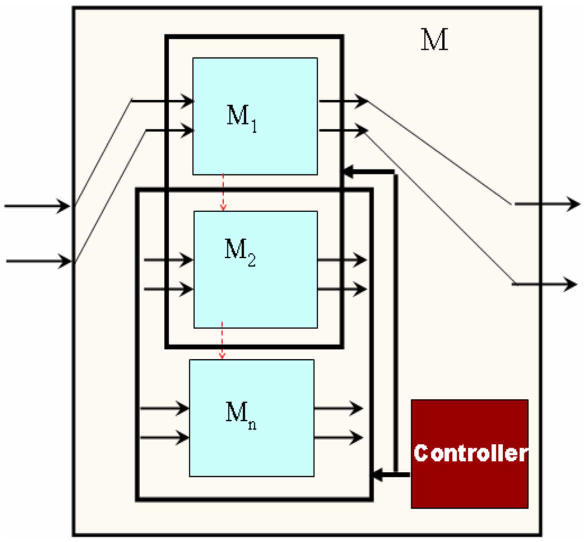


Figure 34 - Multistage Model

Figures 18 and 19 are simple illustrations of multimodel, in which the submodels *M1, M2*, and *M3* are representations of the same entity at different levels of resolution. Figure 2 presents a multi-model, in which there are two subsets of models: *S1= {M1, M2}* and *S2= {M2, M3}*. Only a single set of submodels is active at a given time in a multistage model, unless multisimulation is performed. The controller agent facilitates cross resolution consistency and management of state updates across submodels. The controller agent of a multistage model decides when and how to switch between sets of submodels as the simulation unfolds.

MRMSM encapsulates a set of components/submodels that represent a system or phenomena at different levels of abstraction possibly at multiple phases. Since such an entity can simultaneously operate at multiple levels of resolution, a number of submodels can be active at a time. Shifts in the phase of a problem trigger model updating by changing the active set of entities. For the sake of simplicity we assume that the number of entities needed in a MRMS model is known in advance.

6.1 Requirements

Requirement 1- An MRMSM should be decoupled from the knowledge regarding how its submodels are composed, created, and represented.

This requirement is critical to facilitate that the MRMSM can be configured with one of multiple families of submodels representing an entity at different levels of resolution for a specific stage of the problem. Furthermore, this requirement imposes the constraint that a family of submodels is designed to work together, and that the MRMSM should be independent of how these submodel families are created and composed. This requirement enables isolating implementation of submodels from the MRMSM.

Requirement 2 - The effects of concurrent interactions at multiple levels of resolution must be combined consistently.

MRMSMs constitute multiple submodels that co-exist during simulation. That is, for instance, a low-resolution and high-resolution entity within the model may simultaneously interact with other external entities at corresponding levels of resolution. Consistency problems between different resolution levels might emerge if shared resources are not used mutually exclusively. Concurrent access to such resources needs to be controlled and coordinated.

Requirement 3 – The state of entities at different levels of resolution within a MRMSM should be consistent.

In an MRMSM the state of the aggregated low-resolution entities should be updated as the state of the corresponding high-resolution entities change. This requires definition of multiple one-to-many dependencies between entities so that when one object changes state, its dependents are notified and updated automatically.

Requirement 4 – In an MRMSM the entities should be allowed to alter their behavior when their internal states are changed.

MRMSM behavior depends on the state and/or stage of the problem, and it must change its behavior at run-time depending on that state/problem phase. Therefore, an MRMSM should enable localization of state-specific behavior and partition behavior for different stages. MRMSM should enable the definition of family of related protocols, encapsulate each one, and make them interchangeable to facilitate varying the protocols independently of the entities that are configured with them.

Requirement 5: The constraints regarding when and under what conditions (1) the consistency of the elements of families of submodels in a multiresolution model be enforced and (2) a shift in the stage of the problem be triggered must be independent of the MRMSM.

Corresponding to the time path of the change of a problem should be a time path of the appropriate submodel families. But, the question is what should be the sequence of this shift pattern of models of family of? Or should there be trigger mechanisms indicating when a shift should occur? This requirement suggests that the constraints be separated from the MRMS model and not be intertwined with the entities to facilitate reuse of the submodel families in a different context with different constraints

Requirement 6 - At the time of the model update, the new set of low-resolution and high-resolution entities and their dependent components must be dynamically loaded and linked into the run-time environment of the simulation.

This requires new model and simulator decoupling strategies that avoid persistent connections to facilitate extensibility. Also, the intricate details of complex model instantiation process should be as independent of the MRMSM as possible to enable flexible update.

Requirement 7 - The consistency of entities undergoing (re)placement needs to be preserved.

The event scheduling and simulation protocol need to be restricted or regulated to facilitate interleaving of entity replacement activities with the simulation events.

Requirement 8 - The state of an MRMSM must be constructed, or at least continue from a specific state after an update operation.

This requires externalization through abstraction, state saving, transmission, and reconstruction after the update operation.

Requirement 9 – An MRMSM should have a mechanism for changing the structure and behavior of the model dynamically. In other words, an MRMS model should support its modification a priori.

This requirement suggests that MRMSMs need to provide facilities that establish a self-representation of the model, offer means by which this representation can be updated, and assure that the manipulations to the self-representation influence the behavior of the model.

Requirement 10 – An MRMSM should provide flexibility in terms of defining behavioral resolution to facilitate analysis that are independent of an implementation and programming language specific constructs.

Component-based, function-based, and attribute-based approaches to resolution refinement are course-grain approaches. However, flexible what-if analysis requires analysts to select the behavioral resolution at arbitrary levels of granularity without relying on implementation specific constructs. That is, a high-level *behavioral formalism* that enables *fine-grain stepwise refinement* is needed to specify the behavior of an entity at multiple levels of resolution. This facilitates fine-tuned control of a simulation at run-time at different levels of resolution. By having capabilities to control the scope of update, analysts can perform what-if analyses at arbitrary levels.

Requirement 11 – Multisimulation with multimodels, multiaspect models or multistage models needs mechanisms to decide when and under what conditions to replace existing models with a successor or alternative.

Staging considers branching to other simulation studies in response to scenario or phase change during experimentation. Graphs of model families may facilitate derivation of feasible sequence of models that can be invoked or staged. More specifically, a graph of model families can be used to specify alternative staging decisions. Each node in the graph depicts a model, whereas edges denote transition or switching from one model to another. Figure 20 depicts the components of the abstract architecture of a possible multisimulation engine.

A meta-simulator is a scheduler that generates staged composition of models by traversing the model stage graph and coordinates their simulation and staging within distinct simulation frames. Each frame simulates a distinct subset of models derived from the model stage graph. Note however, not all staged compositions are feasible or useful. Hence, the meta-simulator needs to consult with the model recommender before model staging to determine if the emergent trigger or transition condition in the simulation is consistent with the precondition of the model to be staged. More than one model in a family can qualify for staging; in such cases separate simulation frames need to be instantiated to accommodate and explore plausible scenarios. Given a collection of models (or more generally, family of models), a stage graph can be generated automatically by an optimistic approach that connects every available node (model) to every other node within the domain of problem. The edges in a model stage graph denote plausible transitions between models as the problem shifts from one stage to another.

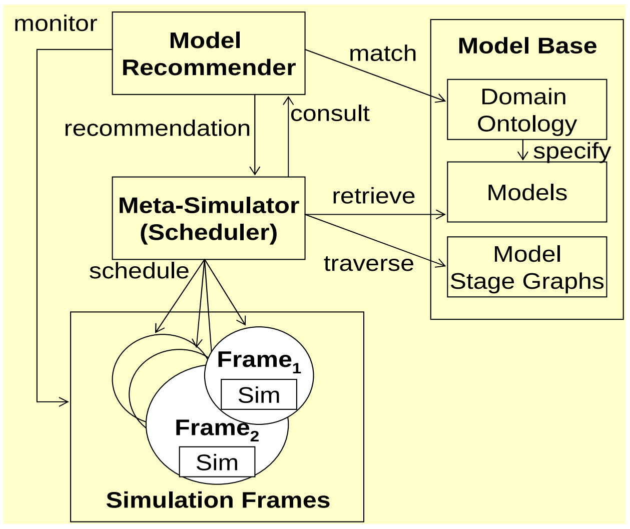


Figure 35 - Components of the Multisimulation Engine

Model recommendation in multisimulation can simply be considered as the exploration of the model staging space that can be computed by reachability analysis of the graph. There are two modes for the usage: (1) Offline enumeration of paths using the graph and performing a staged simulation of each model in sequence one after the other, unless a model staging operation becomes infeasible due to conflict between the transition condition and the precondition of the successor model and (2) run-time generation of potential feasible paths as the simulation unfolds. In both cases, an online model recommender plays a key role to qualify a successor model. The first case requires derivation of sequence of models using a traversal algorithm. The edges relate families of models. Therefore, the actual concrete models, the preconditions of which satisfy the transition condition need to be qualified, since transition to some of these model components may be infeasible due to conflict between a candidate model and inferred situation. Identifying such infeasible sequences is computationally intractable; otherwise, it would have been possible to determine if the conjunction of two predicates is a tautology by using a polynomial time algorithm.

Experience in component-based simulation paradigm, however, indicates that for most model components preconditions are simple. Hence, it is possible to eliminate some models that violate the transition condition. For the remaining possible transitions it is possible to select one of the three strategies: (1) omit all difficult qualification conditions, (2) decide on an edge by edge basis which specific models of a model family to include, and (3) include all difficult edges. Omitting all difficult associations between transitions and model preconditions is conservative. This strategy excludes all infeasible models. The cost is the exclusion of some feasible edges. Hand-selecting those associations between transition conditions and models facilitates inclusion of feasible models. Yet, the costs involved with this level of accuracy are the potential human-error and effort needed to filter out infeasible models. Choosing to include all difficult associations is liberal, in that it ensures inclusion of all feasible models. The cost is the inclusion of some infeasible models, hence the inclusion of some undesirable staged compositions that enforce models to be simulated even when their qualification conditions are violated. Nevertheless, it is possible to screen out such models using an online model recommender.

Candidate models and associated simulations can be maintained by *focus points*. A focus point manages branch points in the simulation frame stack. Suppose that a goal instance (i.e., stage transition condition) is at the top of the stack. If only a single model qualifies for exploration, then it is pushed onto the stack. Yet, if more than one model matches the condition, a simulation focus point is generated to manage newly created simulation branching (discontinuity) points, each one of which have their own contexts. When a path is exhausted, the closest focus point selects the next available model to instantiate the simulation frame or return to the context that generated the focus point. As simulation games are explored, a network of focus points is generated. Determining which focus point should be active at any given time is the responsibility of the meta-scheduler. When more than one model is qualified, then scheduler needs to decide which one to instantiate. Control rules can inform its decision. Three steps are involved in deploying a new simulation frame in such cases: matching, activation, and preference. The matching step should both syntactically and semantically satisfy the request. The activation step involves running a dynamic set of rules that further test the applicability of models with respect to contextual constraints. Finally, the preference steps involve running a different set of rules to impose an activation ordering among the active frames

The question is what particular model design specification approach is conducive to handling the above requirements. For instance, most of the requirements above need facilities to explicitly specify the (1) transition conditions between model families, (2) cross resolution consistency management, and (3) actions needed for dynamic model replacement and seamless configuration update.

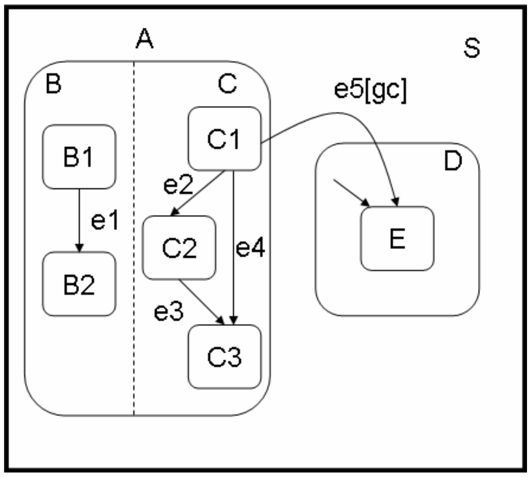


Figure 36 - Multimodel Structure

Figure 21 depicts the structure of a multimodel *S*, which has submodels *A* and *D*. At first glance, the diagram can be interpreted as a statechart. However, in our context each node designates a submodel or model component, whereas arcs denote transitions between models. Each transition indicates the event that triggers the activation of a target node, the conditions under which it is activated, and the actions taken to complete the change in the configuration. Submodel A, in Figure 22, can be a multiresolution model with two concurrent levels, each one of which is represented by submodels *B* and *C*.

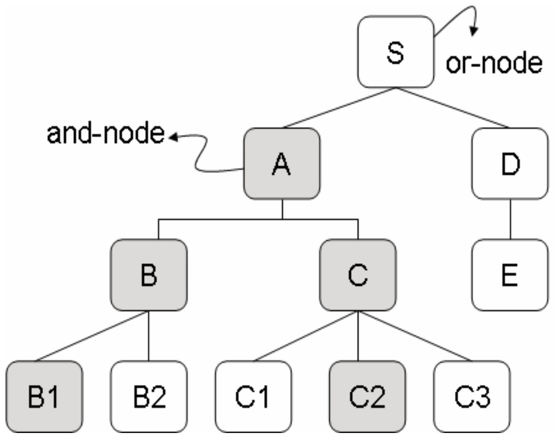


Figure 37 - Multimodel And-Or Graph

Note that the level of decomposition can be arbitrary by extending the hierarchy. Similarly, submodels *B* and *C* can represent two different aspects of the same subsystem represented by model A. The submodel *D* and alternative configurations of models within *A* facilitate representing distinct stages of the problem. The simulation of a multimodel is based on the interpretation of the model reachability structure depicted in Figure 22. The structure is an *and-or graph* that has two types of nodes. The *and-nodes* require every descendant node to be active. On the other hand, only one descendant node of an or-node is active, while the rest are dormant. In Figure 22, the model S is represented as an *or-node*, as either *A* or *D* is active at any point in time during the simulation. The submodel *A*, however, is an *and-node*, as both *B* and *C* can be active. The submodels *B* and *C* are *or-nodes*. This requires either *B1* or *B2* (basic nodes of *B*) to be latent within submodel B. Similarly, at most one of the submodels of C can be active at any point of time.

Definition: For every pair of nodes, their **scope** is the lowest common ancestor, e.g., *scope (B1, E)= S.*

Definition: A pair of nodes is **orthogonal** if their scope is an or-node.

Definition: The **configuration** of a multimodel S is a minimal subtree *CFG(S)* that satisfies

- $S \in CFG(S)$
- If $A \in CFG(S)$ and *A* is an or-node then *CFG(S)* contains exactly one of the submodels of *A*.
- If $A \in CFG(S)$ and *A* is an and-node then *CFG(S)* contains all of the submodels of *A*.

The **basic configuration** is the set of basic nodes in a configuration, which uniquely identifies a multi-model configuration. Simulating a multimodel requires the definition of its state and semantics of the transitions that facilitate the generation of the its behavior. The state of a multimodel consists of the following elements.

- CFG - A configuration with respect to the multimodel under consideration.
- TR - History information – last visited submodel.
- CND- Assignment of all atomic conditions.
- TM – Set of all scheduled timeout events at the current state.
- E – The subset of atomic events.

A transition takes place to change the configuration of a multimodel to switch to a new stage, insert new set of submodels, update the level of resolution and fidelity. A transition, (*M1, <eventExpr, condExpr, actions>, M2*), from model M1 to M2 is triggered or enabled when

- $M1 \in X.CFG$

- $eventExpr \in (X.E \cup X.TM)$
- $X.CND \prec condExpr$

Once a transition is enabled, its execution results in the change of the configuration of the multimodel. To specify the change, we need to introduce the semantics of a transition. Given a transition $t = (R, < E, C, A >, P)$ at a status *X*, let $S = scope(R, P)$. The set of nodes that are exited as a result of the transition is defined as $EXIT(t) = X.CFG(S) - \{S\}$. The set of nodes that depict the submodels that are activated is defined as $ENTRANCE(t) = CFG(S, P)$. This set of nodes refers to the configuration emerging from S (excluding S) and then including *P* and then expanding by initial node labels. When an enabled transition transforms the configuration of a multimodel the following steps take place.

- A new construction is created by replacing the source submodel M1 with the target submodel, M2.
- The entrance conditions associated with M1 (e.g., *in(M1)*) become false, whereas the entrance conditions associated with M2 (e.g., *in(M2)*) become true.
- The events *exit(M1)* and *enter(M2)* are added to the new status.
- The actions specified in the transition are executed. These actions need to address the requirements listed above.

Designing multimodels based on the above formalism in terms of communicating and interacting objects and classes that are customized to deal with the requirements listed above is a significant challenge. The next section illustrates how the principled use of a number of design patterns may facilitate the realization of a selected requirement with the above formalization in mind.

6.2 Design Principles for MRMSM

The specification formalism discussed in the previous section requires composing objects into tree structures to represent part-whole hierarchies. Furthermore, each submodel needs to be treated uniformly regardless of whether it is a composition (e.g., multiresolution) or an individual submodel. Multiresolution or multistage models can be elements of coarse granular models, which operate at multiple levels of resolutions and stages themselves. Hence, it is critical to make sure that a multimodel simulator that uses the strategy discussed above ignores the differences between compositions of submodels and submodels or individual components. The design structure between *Model*, *CompositeSubModel*, and *Submodel* aims to achieve this objective. Specifically, the *Model* defines the interface for objects in the composition, which realize the default behavior. The *Submodel* represents those nodes in the composition that have no children nodes. The *CompositeSubmodel* defines behavior for submodels that have multiple levels of resolution or aspects.

A significant requirement as discussed in the previous section is to *decouple MRMS from the knowledge regarding how its submodels are composed, created, and represented*. The *BehaviorFactory* component facilitates the creation of families of related or dependent submodels or model components without explicitly specifying the concrete realizations. For instance, referring to Figure 22, the structure and behavioral representation of submodel *B* can be encapsulated within the *BehaviorB* component of Figure 23. As a result, the MRMS model (i.e., *Model*) is independent of how submodel *B* is instantiated. This information is encapsulated within *BehaviorB*. That makes it possible configure the MRMS model with one of several submodel families. Furthermore, since the related set of submodels and their components are designed to be used together, the decoupling of MRMS from *BehaviorB* enables enforcing this constraint. Finally, this configuration makes it easy to exchange submodel families easier.

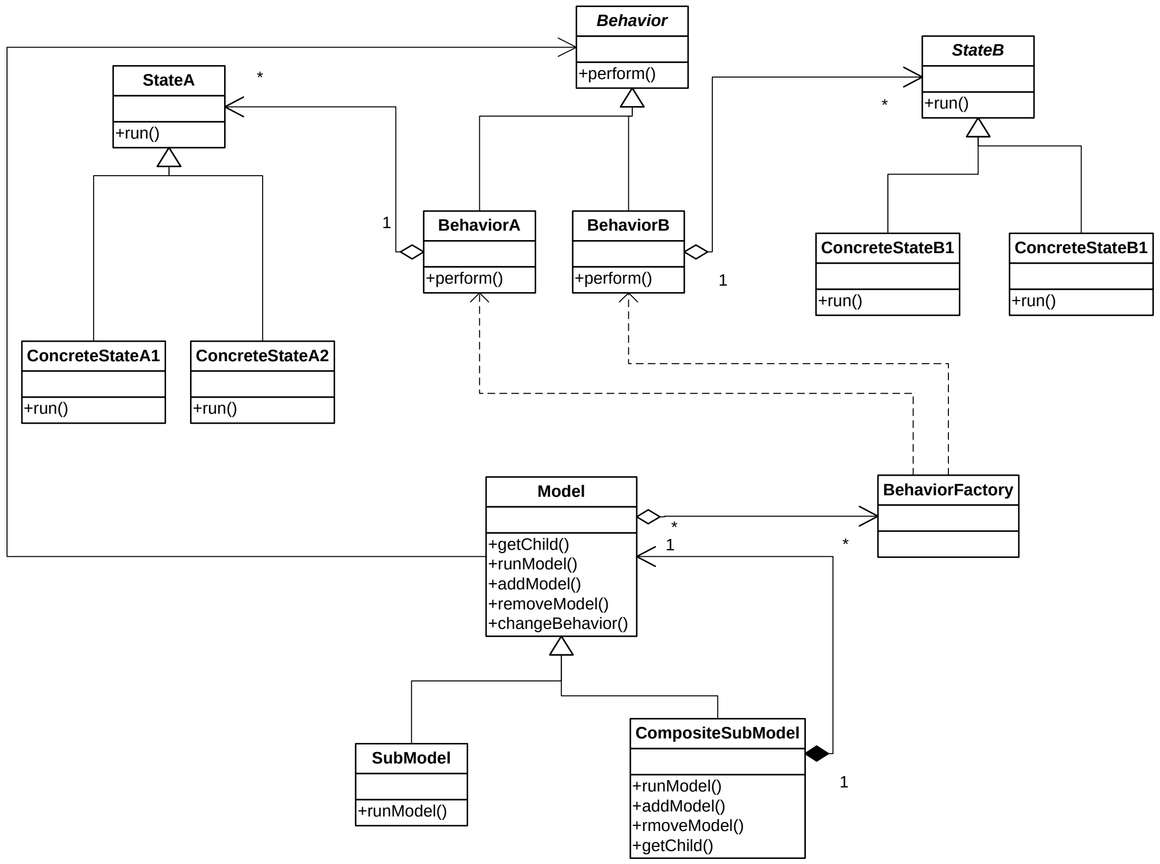


Figure 38 - A Partial and Preliminary Design Strategy for Multimodels

The existence of *State* components associated with *Behavior* entities aims to allow a submodel to alter its behavior when its internal configuration changes (i.e., *submodel C* starts using *C2* as opposed to *C1* after the configuration of the multimodel is updated). As a result, the submodel will appear to change its subcomponents. That is, the submodels behavior depends on the configuration of the multi-model, and the submodel needs to change its behavior at run-time depending on the new configuration. The consequence of this design strategy is that configuration-specific behavior and partitions behavior for different configurations. This makes the model transition concept presented above explicit.

7. Conclusions

The significance of simulation modeling at multiple levels, scales, and perspectives is well recognized. However, existing strategies for developing such models are often application specific. The position advocated in this paper is that generic design principles for specifying and realizing multiresolution, multistage models are still lacking. Requirements for simulation environments that facilitate multiresolution multistage model specification are introduced. A multimodel specification formalism based on graph of models is suggested along with design precepts to enable flexible dynamic model updating. The notion of multisimulation is introduced to enable exploratory simulation using various types of multimodels.

8. References

Banks J.S. and Sundaram RK (1990) Repeated games, finite automata, and complexity. Games and economic behavior, vol 2, pp. 97–117

BISC-SIG (2004). Special Interest Group in Anticipatory Systems. BISC (Berkeley Initiative in Soft Computing) Available online via <http://www.anticipation.info/> (accessed on April 9, 2004)

Bloom A (1968) The republic of Plato (translated with notes and an interpretive essay). Basic Books Inc., New York

Cannon-Bowers J, Salas E and Converse S (1993) Shared mental models in expert team decision making. In Castellan NJ (Ed) Individual and Group Decision Making: Current Issues, Hillsdale, NJ: Erlbaum, pp 221–246

Dacey R and Carlson JL (2004) Traditional decision analysis and the poliheuristic theory of foreighn policy decision making. Journal of conflict resolution, vol 48, no 1, pp 38-55

Daly JJ and Tolk A (2003) Modeling and Simulation Integration with Network-Centric Command and Control Architectures, Fall Simulation Interoperability Workshop, IEEE CS Press

Davis KP, Kulick J and Egner M (2005) Implications of modern decision science for military decision-support systems. RAND

Endsley MR (1995) Measure of situation awareness in dynamic systems. Journal for Human Factors, vol 37 no 1. pp 65-84

Eisenhardt KM and Zbaracki JM (1992) Strategic decision making. Strategic management journal, vol 13, pp 17-37

Evan JR and Olson DL (2002) Introduction to Simulation and Risk Analysis (2nd Edition), Prentice Hall, pp 161–179

Fraser MN and Hipel WK (1984) Conflict analysis models and resolutions. Elsevier Science Publishers

Gaba DM and Howard SK (1995) Situation awareness in anesthesiology. Journal for Human Factors, vol 37, pp 20-31

Geyer F and van der Zouven J (1998). Bibliography on sociocybernetics (3rd edition). (http://www.unizar.es/sociocybernetics/bibliografia.html, 1998)

Hamm RM (1988) Moment by moment variation in expert's analytic and intuitive cognitive activity. IEEE transactions on systems, man, and cybernetics, vol 18, no 5, pp 757-776

Hammond KR (1986) A theoretically-based review of theory and research in judgement and decision making. Technical report CRJP 260, Center for Research on Judgement and Policy, University of Colorado

IEEE 1516-2000 Family of Standards for Modeling and Simulation (M&S) High Level Architecture (HLA) – (a) IEEE 1516-2000 Framework and Rules; (b) IEEE 1516.1-2000 Federate Interface Specification; (c) IEEE 1516.2-2000 Object Model Template (OMT) Specification

Kahan JP and Rapoport A (1984) Theories of coalition formation, Hillsdale, NJ: Lawrence Erlbaum Associates, 1984

Klein G (1997) The recognition-primed decision (RPD) model: Looking back, looking forward. In Zsambok EC and Klein G (eds) Naturalistic Decision Making, Lawrence Erlbaum Associates Publishers, pp 285-292

Knight DE, Curtis WH and Fogel LJ (eds) (1991) Cybernetics, simulation, and conflict resolution, Proc. of the 3rd annual symposium of the American Society for Cybernetics (Spartan Books, New York)

Leimar O (1997) Repeated games: A state space approach. Journal of Theoretical Biology, vol 184, pp 471–498

Murch R and Johnson T (1998). Intelligent Software Agents. Prentice Hall

Mintz A (2004) Decision making in familiar and unfamiliar settings: An experimental study of high-ranking military officers. Journal of conflict resolution, vol 48, no 1, pp 49-62

Niland M. W. (2006). The migration of a collaborative UAV testbed into the FLAMES simulation environment. In Proceedings of the 2006 Winter Simulation Conference, pp. 1266-1272.

Ören TI (1987). Model update: A model specification formalism with a generalized view of discontinuity. In Proceedings of the summer computer simulation conference, Montreal, Quebec, Canada, 1987 July 27-30, 689-694

Ören TI (1991) Dynamic templates and semantic rules for simulation advisors and certifiers. In Fishwick AP and Modjeski BR (eds) Knowledge-based simulation: Methodology and application, Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, pp 53-76

Ören TI (2001) Towards a modelling formalism for conflict management. In Sarjoughian SH and Cellier EF (eds.) Discrete-event modeling and simulation: A tapestry of systems and AI-based theories and methodologies, Springer-Verlag, New York, pp 93-106

Parsons S and Wooldridge M (2002) An introduction to game theory and decision theory. In Parson et. al (eds) Game theory and decision theory in agent-based systems, Kluwer Academic Publishers, pp 1-28

Power D (2002) Decision Support Systems: Concepts and Resources for Managers; Chapter 10: Building Model-Driven Decision Support Systems, Quorum Books division Greenwood Publishing, pp. 172ff

Rosen R (1985) Anticipatory systems – Philosophical, mathematical and methodological foundations. Pergamon Press, New York

Sallach D (2003). Interpretive agents. In Proceedings of the Agent2003: Challenges in Social Simulation Conference. University of Chicago

Serfaty D, Macmillan J, Entin EE and Entin BE (1997) The decision-making expertise of battle commanders. In Zsambok EC, Klein G (eds) Naturalistic Decision Making, Lawrence Erlbaum Associates Publishers, pp 233-246

Shubik M (ed) (1964). Game theory and related approaches to social behavior. Wiley, New York

Simon H (1982) Models of bounded rationality, vol 1, Cambridge, MA, MIT Press

Silvermann GB (1994) Unifying expert systems and the decision sciences. Operations research, vol 42, no 3, pp 393-413

Sokolowski, JA (2003) Enhance Decision Modeling Using Multiagent System Simulation, SIMULATION, vol 79 no 4, pp. 232-242

Stanovich EK and West AR (2002) Individual differences in reasoning: Implications for the rationality debate. Behavioral and brain sciences, vol 23, no 5, pp 645-726

Tolk A and Kunde D (2003) Decision Support Systems in the Military Environment., In: Tonfoni G. and Jain L. (Eds.) Innovations in Decision Support Systems, International Series on Advanced Intelligence, Advanced Knowledge International, Magill, Adelaide, Australia, pp 175-210,

Tolk A (2004) An Agent-based Decision Support System Architecture for the Military Domain. In: Phillips-Wren GE and Jain LC (eds) Intelligent Decision Support Systems in Agent-Mediated Environments, Volume 115 Frontiers in Artificial Intelligence and Applications, IOS Press, pp 187-205

Tolk A and Gaskins RC (2006) Challenges and Potential of Service-oriented Architectures for Net-Centric Operations. NATO Report MP-HFM-136, Research and Technology Organization: Human Factors and Medicine Panel, Brussels

Tversky A and Kahneman D (1974) Judgment under uncertainty: Heuristics and biases. Science vol 185, pp 1124-1131

von Neumeann J and Morgenstern O (1953) Theory of games and economic behavior. (3rd ed) Princeton, NJ, Princeton University Press

Wikipedia: (Phi-Per) (2004) Philosophy of Perception. Available online via <http://en.wikipedia.org/wiki/Philosophy_of_perception>

Wooldridge M (1999) Intelligent agents, In Weiss G (ed) Multiagent systems: A modern approach to distributed artificial intelligence, Cambridge, Massachusetts, The MIT Press pp 27-77

Wooldridge M (2002) An introduction to multi-agent systems. John-Wiley & Sons, Ltd

Yilmaz L and Ören T (2004) Dynamic model updating in simulation with multimodels: A taxonomy and generic agent-based architecture. In Proceedings of the SCSC'04. pp. 3-8

Yilmaz L and Ören T (2006) Simulation-Based Problem Solving Environments for Conflict Studies: Toward Exploratory Multisimulation with Dynamic Simulation Updating. Simulation and Gaming Journal (forthcoming)

Zachary W (1998) Decision-support systems: Designing to extend the cognitive limits. In: Helander GM (ed) Handbook of Human-computer interaction. Amsterdam: Elsevier Science Publishers

Zeigler PB., Praehofer HP and Kim GT (2000). Theory of modeling and simulation: Integrating discrete event and continuous complex dynamic systems, Academic Press

Zsambok EC (1997) Naturalistic decision making: Where are we now? In Zsambok EC and Klein G (eds) Naturalistic Decision Making, Lawrence Erlbaum Associates Publishers, pp 3-16